[20] Multiple Protein Sequence Alignment: Algorithms and Gap Insertion

By WILLIAM R. TAYLOR

Introduction

The basic method of sequence alignment has remained unchanged for over 20 years. In that time the most significant development has been to add a termination condition to display local alignments; all other algorithms and modifications have been concerned with increasing speed, the effect of various gap penalties and relatedness matrices, or extensions for multiple sequence data. This status quo has remained despite results in which it is clear to the eye that the alignments produced sometimes do not reflect the expected biological pattern of regions of weak matches (including gaps) and regions of good matches. This characteristic of the data has not been ignored, but has been dealt with by methods other than conventional dynamic programming (typically pattern matching). Within a dynamic programming approach, failure to attain the desired alignment has usually been attributed to an incorrect combination of gap penalty and relatedness matrix; however, the underlying problem is more fundamental and some novel (and revived) algorithms are described below that have been developed by the author in an attempt to overcome these defects.

Outline of Algorithms

Score Run-Length Enhancement

Consider the following simple example: the sequence ABQDDEFHRSKKLMO, when matched in phase with the alphabet, gives rise to six scattered matches. However, if displaced back by one position two segments (DEF and KLM) align, also giving six matches. With real sequences, the trained eye, of one who is familiar with remotely related protein sequence alignments, might suspect that a scattered score distribution is less biologically significant than a clumped distribution. This quality in a sequence alignment that makes it look correct has been quantified and turned into a method to improve basic pairwise and multiple sequence alignment.[1]

[1] W. R. Taylor, *J. Comput. Biol.* **1,** 297 (1994).

Profile Gap Weighting

In the development of a multiple sequence alignment profile, gaps can accumulate and make the profile longer than the average sequence length and even longer than any sequence that it contains. This elongation creates a bias for the profile to recognize longer members of the family as these require less insertions to be aligned and hence incur less penalty. An algorithm is described that prevents this undesirable behavior by comparing the average distance (excluding gaps) between pairs of positions and using this difference to modify the gap penalty.[2]

Gap Bias by Structure Prediction

The use of known and predicted secondary structure in sequence matching, while frequently examined, has not been fully exploited. For the insertion of gaps, previous methods considered only the opening of a gap at a single point and not what is contained in the gap once opened or what terminates the broken sequence ends. Incorporating all these aspects is computationally more difficult (and slower) but makes a difference both to the stability of the correct alignment and to the detailed placing of gaps.[3]

Score Run-Length Enhancement

Background

Sequence alignment has revealed many unexpected and often important similarities that have given insight into previously obscure systems. Underlying much of this success is the widely used dynamic programming (DP) algorithm[4-6] which finds the optimal alignment of the sequences under a given scoring scheme. Although the algorithm is rigorous, the parameters are poorly characterized. These include the model relatedness between sequence elements (either residues or nucleotides) and the penalty for gaps. Despite much analysis, there is still little to guide the best choice of these.

Multiple sequence alignment typically identifies strongly scoring regions (where gaps are less frequent), which, in situations where the structure is known, are often found to correspond to core secondary structures (or motifs). With only a pair of sequences, introducing correlation between

[2] W. R. Taylor, *Bull. Math. Biol.* in press (1996).
[3] W. R. Taylor, *Gene* **165**, GC27 (1995).
[4] S. B. Needleman and C. D. Wunsch, *J. Mol. Biol.* **48**, 443 (1970).
[5] P. H. Sellers, *J. Combinator. Theor.* **16**, 253 (1974).
[6] T. F. Smith and M. S. Waterman, *J. Mol. Biol.* **147**, 195 (1981).

adjacent matches should emulate this uneven distribution of score, enhancing the score for well-matched regions, not only for the matching elements but also for the surrounding region to an extent expected for the size of a typical core structure. The following method describes how this can be achieved, dealing with the problems of correlation length and score normalization. For simplicity, the method is sometimes referred to below as the motif-bias method.

Algorithm

Basic Dynamic Programming. Dynamic programming compares sequences by finding the best path through a matrix of scores from all pairwise matches of elements between the two sequences. This is achieved by incrementally extending each path with a locally optimal step. Element $d_{i,j}$ in the score matrix can extend any path from the preceding row or column to produce the highest path score. Applying this condition to each matrix element transforms the pairwise score matrix into a matrix of path scores. This can be represented recursively:

$$s_{i,j} = d_{i,j} + \max \begin{cases} s_{i-1,j-1} & \\ s_{i-1,m} - g & (m < j - 1) \\ s_{n,j-1} - g & (n < i - 1) \end{cases} \tag{1}$$

where g is a gap penalty. Path connectivity is recorded in a matrix of pointers, and if all d_{ij} are positive, then the path from the highest score is the optimal (global) alignment.

Modified Algorithm. The modified DP algorithm should give higher scores for runs of good matches. This can be achieved by accumulating a running product of match scores and giving a higher score to long runs of matches relative to more scattered matches.

Running-product score. The effect of the running-product score depends on the match score values: allowing a zero score, the product will disappear and not recover afterward; whereas if the minimum is one, the product will grow continually. It is therefore necessary to rescale the match scores into a reasonable range, which is equivalent to continually damping by a constant value (a):

$$r_i = (r_{i-1} + 1)(d_i + 1)/a \tag{2}$$

where d_i is ith in the series $\{d_1 \ldots d_N\}$ (with $d_i \geq 0, \forall_i$) and r_i is the current product.

Automatic damping. Real (as distinct from random) sequences allow the possibility that very similar sequences will give astronomic products unless the value of a is chosen suitably large in anticipation. This problem was avoided by defining a in terms of the final score:

$$a^n = r_N/N + \bar{d} \tag{3}$$

where r_N is the terminal product in a sequence length N and $\bar{d}$ the average score (n is a parameter). Defining a as a function of itself means that a solution for its value must be found by iteration, and to give stability to this convergence, a further constant c was introduced:

$$r_i = (d_i + 1)(r_{i-1} + 1)/(ac) \tag{4}$$

A good starting estimate of a was $a^n = \bar{d} + 1$.

Dynamic programming formulation. To include gaps, the above scoring function must be incorporated into the DP algorithm, and, in keeping with the overall goal, the running product [Eq. (4)] should not jump across gaps:

$$r_{i,j} = \begin{cases} (d_{i,j} + 1)(r_{i-1,j-1} + 1)/(ac) \\ 0, \quad \text{after gaps} \end{cases} \tag{5}$$

In Eq. (5), match scores are now elements of a matrix.

To produce the desired property that the score for good sequence matches tends toward the unbiased score, the unmodified sum [Eq. (1)] was combined with the running product into a sum t:

$$t_{i,j} = d_{i,j} + \max \begin{cases} t_{i-1,j-1} + (d_{i,j} + 1)(r_{i-1,j-1} + 1)/(ac) & : r_{i,j} = (d_{i,j} + 1)(r_{i-1,j-1} + 1)/(ac) \\ t_{i-1,m} - g \quad (m < j - 1) & : r_{i,j} = 0 \\ t_{n,j-1} - g \quad (n < i - 1) & : r_{i,j} = 0 \end{cases} \tag{6}$$

As r_{ij} depends on the maximum in Eq. (6), its conditional assignment is shown on each line of the equation. Designating the expression left of the ":" as *left* and that right of the "=" as *right*, this construct reads: "**if** the $|left|$ is maximum **then** assign r_{ij} the $|right|$" (for $|expression|$, read "the value of the *expression*").

Application

All pairwise alignments of 12 remotely related aspartyl proteases were calculated and assessed by the percentage of correctly aligned motifs for different gap penalty g and similarity matrix m. (See Ref. 1 for details.)

Unbiased Alignment. The results for the unbiased method showed little sensitivity to the form of the similarity matrix but were strongly dependent on the gap penalty. Above $g = 10$, accuracy dropped from around 70–75% toward 40–50% at $g = 30$ (Fig. 1a).

Motif-Biased Alignment. Using the motif-biased algorithm [Eq. (6)], different parameter values for n [Eq. (3)] and c [Eq. (5)] produced different behavior. However, almost all combinations extended the region of good accuracy (better than 65%) into higher gap penalties, and with $n = 3$, $c = 5$ this accuracy was obtained up to $g = 30$. In addition, the region of highly gapped alignments (over 10 gaps) contracted slightly (Fig. 1b).

Gap-Normalized Controls. The scores with the motif-biased method will always be greater than those with the unbiased method. To allow for this, the scores were normalized by the number of gaps. The line of 10 gaps is roughly the same between Fig. 1a and Fig. 1b, implying that they are comparable. If fewer gaps were taken then the plots would need to be rescaled. For five gaps this factor is 1.4, but even after this rescaling, the accuracy of the motif-biased method at $g = 30$ still exceeds the unbiased results by 5–10%.

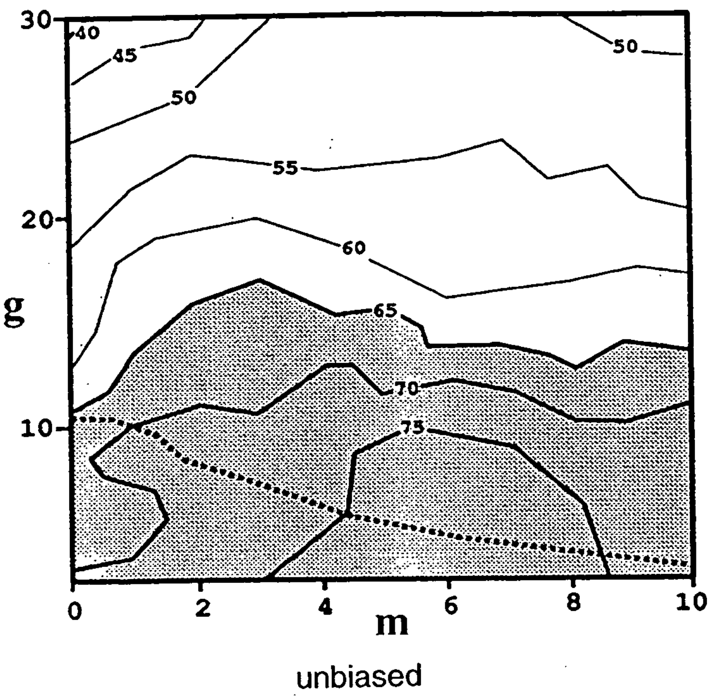
Multiple Alignment. The method was applied to multiple sequence alignment as implemented in the program MULTAL.[7,8] On the basis of results for the pairwise alignments, parameters $n = 3$, $c = 5$, $m = 3$ (Fig. 1b) were fixed and just the gap penalty (g) varied. Using motif bias on aspartyl protease sequences, the correct alignment was obtained over the range $g = 14$–16; by contrast, no correct alignment was found with the unbiased method.

From experience with hierarchic profile clustering, better results were expected from profile/profile matching using a greater weight on identity matches in the later alignment cycles. This was implemented by gradually reducing m with each cycle (moving toward identity matching). With the unbiased method a stable range (gap penalty 11–15) was now found. Using the motif-bias method (with $n = 3$), however, this was extended to 10–22. (Table I).

Alignment Sensitivity. The sensitivity of the modified algorithm was evaluated by comparing all pairwise scores within a family to those obtained between families. For this the aspartyl proteases (Dp) were compared to an equivalent selection of immunoglobulin (Ig) sequences. An average alignment score was obtained within each family (Dp/Dp, Ig/Ig) and expressed relative to the average interfamily (Dp/Ig) score (Table II). With protease data the unbiased method achieved 15% separation of averages but managed only half this with immunoglobulin sequences. Using the biased method, continual improvement in separation was found with in-

[7] W. R. Taylor, *J. Mol. Evol.* **28,** 161 (1988).
[8] W. R. Taylor, this series, Vol. 183, p. 456.

a



b

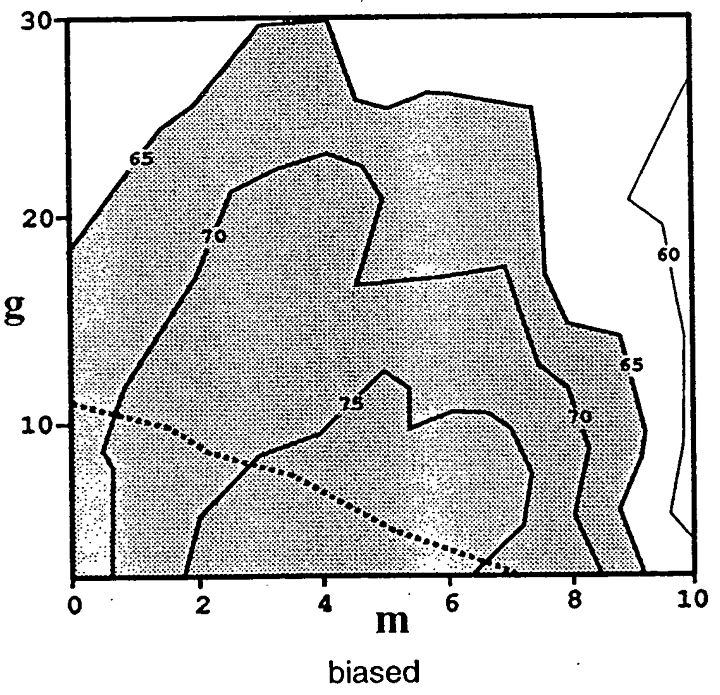


FIG. 1. Accuracy plot varying gap penalty and matrix. The percentage of correctly aligned motifs is plotted for varying gap penalty g against matrix composition m ($m = 0 =$ identity matrix, $m = 10 =$ Dayhoff matrix) for the unbiased method (a) and for the biased methods (b) with $n = 3$ and $c = 5$. Both plots were sampled at intervals of $g = 3$ and $m = 1$ (excluding the region $g < 3$). The area below the dotted line contains alignments with an excessive number of gaps (more than 10).

TABLE I

MULTIPLE ALIGNMENT STABILITY UNDER GAP PENALTY VARIATION[a]

	Gap penalty
	----5--- 10---15- --20--- 25---30---35
	....:..... | ....:.... | ....:.... | ....:
Unbiased	-------+++|||||++------------------
Biased (n = 3)	--------+||||||| ||||||++-----------

[a] At each value of the gap penalty a "|" indicates that an alignment was obtained that correctly aligned the motifs across all sequences. A "+" indicates an incorrect alignment (even if only by one motif in one sequence), whereas a "-" indicates that no test was made.

creasing n, reaching maximum at $n = 5$ with 13% for the immunoglobulins and 20% for the proteases.

Relationship to Other Approaches

Method of Vingron and Argos. Segment matching has been used by Argos and co-workers,[9,10] allowing high-scoring segment combinations to be selected manually or semiautomatically.[11] Their segments have a limited range of fixed lengths, and although this is sufficient for practical purposes, the approach has the theoretical problem that the segments must be scored prior to concatenation, which does not fully exploit the synergism in a number of consecutive weak segments. An equivalent approach has been used for predicting transmembrane segments.[12]

Method of Boswell and McLachlan. The approach of Boswell and McLachlan[13] is similar to the current method in its use of exponential damping to enhance local features. They used a running sum (as distinct from a product in the current method), resulting in less extreme values and so avoiding the complexities of score normalization. They also used a second reverse pass across the matrix to balance lag effects in the score response. Using a product, this is less critical since a product has a sharper response and allows a single pass to be used in the current method. Boswell and McLachlan used their approach mainly to look for local alignments, again allowing them to be less concerned about the score normalization that must be considered with a global method.

[9] P. Argos, *J. Mol. Biol.* **193**, 385 (1987).
[10] P. Argos and M. Vingron, this series, Vol. 183, p. 352.
[11] R. Rechid, M. Vingron, and P. Argos, *CABIOS* **5**, 107 (1989).
[12] D. T. Jones, W. R. Taylor, and J. M. Thornton, *Biochemistry* **33**, 3038 (1994).
[13] D. R. Boswell and A. D. McLachlan, *Nucleic Acids Res.* **12**, 457 (1984).

TABLE II
ALIGNMENT SENSITIVITY UNDER GAP PENALTY (g) VARIATION FOR PROTEASE AND IMMUNOGLOBULIN SEQUENCES[a]

	g for protease				g for immunoglobulin			
n	15	20	25	30	15	20	25	30
0	15.0	15.1	14.7	14.3	7.5	7.5	7.1	6.5
2	16.7	16.9	16.8	16.5	6.9	7.2	7.3	7.3
3	18.7	18.8	19.0	18.8	9.3	9.7	9.9	10.0
4	19.4	19.6	19.5	19.3	10.6	11.1	11.3	11.3
5	20.0	20.4	20.4	20.1	12.4	13.0	13.0	13.3

[a] With and without the motif bias ($n > 1$ and $n = 0$, respectively). The parameter n which controls the emphasis on the run bias was introduced in Eq. (3), and the associated parameter c [Eq. (4)] was assigned the value $n + 2$. The table values are the relative percentage separations of the mean intrafamily score over the mean interfamily score.

Method of Huang. Huang[14] has described a simple enhancement for matched elements in ungapped runs. The sequences on which this method was tested, however, were all more closely related than those for which the current method was developed. Consequently, the main difference in the methods is the degree of bias imposed, which, being relatively mild in the Huang method, avoided the complexities of score normalization.

Profile Gap Weighting

Background

Accumulation of gaps is a problem with profiles, both when aligning a profile with another profile and when aligning with a single sequence. Gaps make the profile longer than the mean sequence length, even leading to a profile longer than any sequence in the alignment. Containing the problem with strong gap penalties can lead to incorrect alignments and is therefore not an ideal solution. Previously, pragmatic steps have been taken to avoid this problem, including excision of variable regions,[15] and altering the gap

[14] X. Huang, *in* "Combinatorial Pattern Matching, Volume 807 of Lecture Notes in Computer Science" (M. Crochemore and D. Gusfield, eds.), p. 54. Proceedings, 5th Annual Symposium, CPM. Springer-Verlag, Berlin, 1994.

[15] J. D. Thompson, D. G. Higgins, and T. J. Gibson, *CABIOS* **10,** 19 (1994).

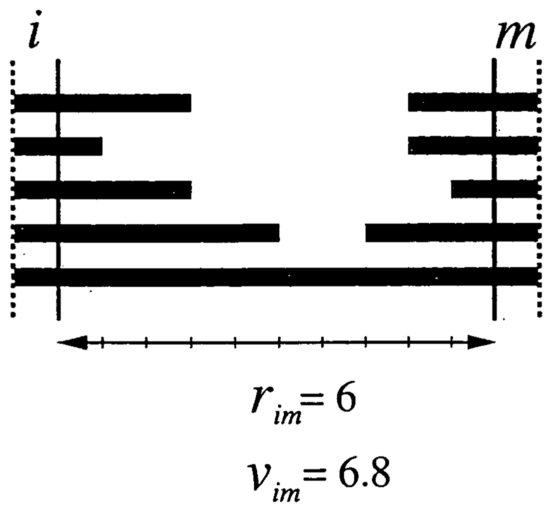


FIG. 2. Sequence profile position separation. Five protein sequences are shown schematically (black bars) aligned into a profile containing gaps (white space). Ten positions separate positions i from m in the profile, but the mean number of intervening residues (r_{im}) is only six (with a variance, v_{im}, of 6.8).

penalty locally.[16–19] Although the former approach obviously solves the problem, it does so with the undesirable loss of data. The second approach is not a solution but only moderates the problem by reducing the penalties: clearly, a sequence that is greater than average length but equal in length to the profile will experience no penalties.

The method described in this section is a generalization of an approach used in matching sequence templates.[20] In that application, discrete patterns embodied a consensus sequence description, and gaps were constrained not only between adjacent templates but also between all pairs (with a weight that reflected the degree to which the ideal separation would be attained). Reapplied at the residue level, the ideal separation becomes the mean separation (excluding gaps) and the target weight, the variance (Fig. 2).

Implementation with combinatorial matching was simple since each template location was determined before the pairwise gap function was evaluated. Within dynamic programming, however, the alignment cannot be calculated independently of the gap penalties. This circularity could be overcome iteratively; however, a single-pass approximation is presented

[16] A. M. Lesk, M. Levitt, and C. Chothia, *Protein Eng.* **1,** 77 (1986).
[17] G. J. Barton and M. J. E. Sternberg, *Protein Eng.* **1,** 89 (1987).
[18] K. Masaharu, F. Kishimoto, Y. Ueki, and H. Umeyama, *Protein Eng.* **2,** 347 (1989).
[19] J. D. Thompson, D. G. Higgins, and T. J. Gibson, *Nucleic Acid Res.* **22,** 4673 (1994).
[20] W. R. Taylor, *Prog. Biophys. Mol. Biol.* **54,** 159 (1989).

below and the implied danger of an asymmetric solution ignored. This potential problem has been fully evaluated[2] and is discussed further in the concluding discussion.

Algorithm

Length-Difference Function. In two aligned profiles **a** $(a_1 \ldots a_M)$ and **b** $(b_1 \ldots b_N)$, the pairs of positions a_i,b_j and a_m,b_n $(i < m,\ j < n)$ have been matched. If the mean number of residues (excluding gaps) between i and m is ${}^a r_{im}$ and that between j and n is ${}^b r_{jn}$, then the length difference $({}^{ab}d)$ is

$$ {}^{ab}d_{ij,mn} = {}^a r_{im} - {}^b r_{jn} \tag{7}$$

A negative difference means ${}^b r_{jn}$ is too long and further gaps in **b** should be discouraged; similarly, with positive d, gaps in **a** should be avoided. This could be implemented using d to alter the gap penalty (g) and can be inversely weighted by the variance (v), giving a modified gap penalty (g') as

$$ g'_{ij} = g + {}^{ab}d_{ij,mn}/({}^a v_{im} + {}^b v_{jn} + c) \tag{8}$$

In Eq. (8), c moderates damping sensitivity and must be greater than zero.

Alignment Path Sum. The DP algorithm requires a decision for every element in the score matrix on whether to match, insert, or delete. This should be influenced not only by a single separation (as above) but by all separations over the current alignment. If the current matrix position is $\{i,j\}$ and q designates an alignment, then the path of q can be specified as ${}^q\mathbf{p}_{ij} = \{{}^q(n,m)_1,\ {}^q(n,m)_2,\ \ldots,\ {}^q(n,m)_K\}$. A gap score $({}^q s_{ij})$ based on the separation differences (d) over the path ${}^q\mathbf{p}_{ij}$ can then be calculated by modifying Eq. (8) to

$$ {}^q s_{ij} = \frac{t}{10} \sum_{k=1}^{K} \frac{|{}^{ab}d_{ij,{}^q m_k {}^q n_k}|}{k({}^a v_{i,{}^q m_k} + {}^b v_{j,{}^q n_k} + c)} \tag{9}$$

with ${}^q m_k, {}^q n_k$ locating the path at node k. The parameters t and c control the magnitude and sensitivity of the bias. Each contribution is normalized by its path length (k).

Dynamic Programming Formulation. Use of the gap scores to modify the gap penalty directly [suggested by Eq. (8)] is not ideal, as this neglects the fact that extension with no indel also has a gap score. To allow for this, the score for the straight (nonindel) extension was subtracted from those involving an indel. This leaves the score of any path without indels equal to the score obtained with the unmodified algorithm.

From the current matrix element, let x be a path with an immediate insertion, y a path with an immediate deletion, and z the straight (diagonal)

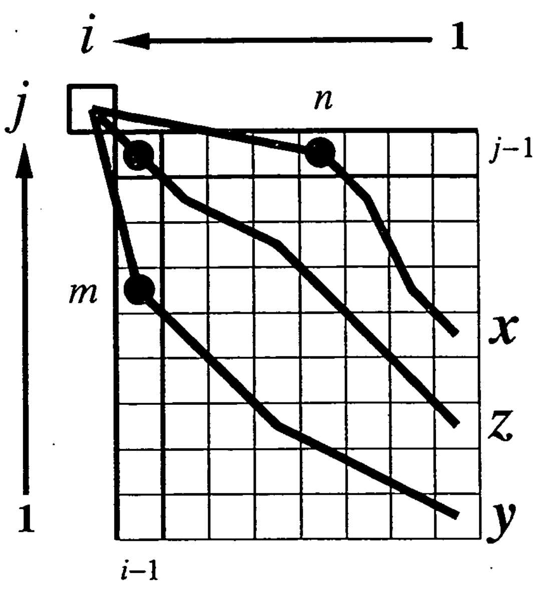


FIG. 3. Dynamic programming with trace back. The calculation of the $\{i,j\}$th element of the score matrix ($e_{i,j}$) requires the values of all elements $\{n < i,\ m < j\}$ (represented by the grid). Conventionally, the path to $\{i,j\}$ is extended from the cell with the maximum value of $\{e_{i-1,j-1}, e_{i-1,m} - g, e_{n,j-1} - g\}$ (g is the gap penalty); however, in the modified algorithm, the optimal alignments from each of these positions (designated z, y, and x, respectively) are also considered. These alignments are specified as a series of paired integers, and in the graph (where $i = 9$ and $j = 10$) the path of alignment y, for example, is ${}^{y}\mathbf{p}_{9,10} = \{(1,1),(3,5),(8,6)\}$.

path with no immediate indel (Fig. 3). The gap scores for each can be used in the DP algorithm as follows:

$$e_{i,j} = f_{i,j} + \max \begin{cases} e_{i-1,j-1} & \\ e_{i-1,m} - g - {}^{y}s_{ij} + {}^{z}s_{ij} & (m < j - 1) \\ e_{n,j-1} - g - {}^{x}s_{ij} + {}^{z}s_{ij} & (n < i - 1) \end{cases} \qquad (10)$$

where, e_{ij} is the element being calculated, f_{ij} the contribution from sequence positions i,j (typically from a relatedness matrix), and g the gap penalty.

The additional requirement over the conventional DP algorithm is the extraction of all alignments for each element (this normally happens only once at the end). This step only requires tracing chains of precalculated pointers, but, compared to just finding the largest of three numbers, it imposes considerable additional burden. To reduce this, a so-called greedy algorithm was implemented in which m and n are preselected to correspond with maximum e:

$$e_{i,j} = f_{i,j} + \max$$

$$\begin{cases} e_{i-1,j-1} & \\ e_{i-1,w} - g - {}^{y}s_{ij} + {}^{z}s_{ij} & (e_{i-1,w} = \max\{e_{i-1,m}, m < j - 1\}) \\ e_{u,j-1} - g - {}^{x}s_{ij} + {}^{z}s_{ij} & (e_{u,j-1} = \max\{e_{n,j-1}, n < i - 1\}) \end{cases} \qquad (11)$$

This requires evaluation of only three alignments per element (compared to $i + j - 1$) and so retains overall quadratic time dependence (given equal-length profiles). Most importantly, it has been shown to make no significant difference in the results.[2]

Application

Data and Scoring. Three families, aspartyl protease, globin, and SH3 domains, were taken as test data. Each included representatives of known structure, and all members within each family were very distantly related. As previously, alignments were assessed by aligned characters in predefined motifs (see Ref. 2 for details). These were summed pairwise over all aligned sequences and normalized by the number of pairs.

Globins. Globin sequences were aligned initially with the unmodified algorithm using a variety of values for the parameters g [gap penalty, Eq. (9)] and m (matrix softness). Of 110 alignments, 11 gave perfect results. When the modified algorithm was similarly tested no improvement was found (measured by the area containing the correct alignment), despite tests over a variety of values for the parameters t and c [Eq. (9)] (Table III). This lack of improvement might be attributed to the few number of gaps required to align the globins. Detailed examination of the alignments indicated that much of the error derived from the misalignment of one very remotely related (bacterial) sequence.

Acid Proteases. The area of the correct alignment for the proteases (Table IV) in $\{g,m\}$ space was similarly investigated for a variety of combinations of t and c values. In contrast to the globins, improvements were found

TABLE III
GLOBIN CORRECT ALIGNMENT AREAS[a]

	t				
c	1	3	5	7	9
1	4	1	0	0	0
5	9	6	4	3	2
10	10	9	7	4	4
15	11	8	7	7	1

[a] The area of the correct globin alignment in the parameter space of m (relatedness matrix) and g (gap penalty) is shown for combinations of the parameters t and c. The ummodified algorithm had an area of 11.

TABLE IV
PROTEASE CORRECT ALIGNMENT AREAS[a]

	t							
c	1	3	5	7	9	11	13	15
1	12	15	15	12	9	8	3	7
5	14	12	15	15	16	20	15	16
10	14	15	13	14	13	13	16	14
15	14	14	9	12	12	11	12	15

[a] The area of the correct protease alignment in the parameter space of m (relatedness matrix) and g (gap penalty) is shown for combinations of the parameters t and c. The unmodified algorithm had an area of 13.

over half the $\{t,c\}$ parameter space, with the largest occurring with high trace-weight (t) (Fig. 4).

SH3 Domains. With SH3 sequences, the correct alignment (which is highly gapped) was never obtained with the unmodified algorithm, whereas with the modified algorithm some correct alignments were found.

Structural Bias for Gaps

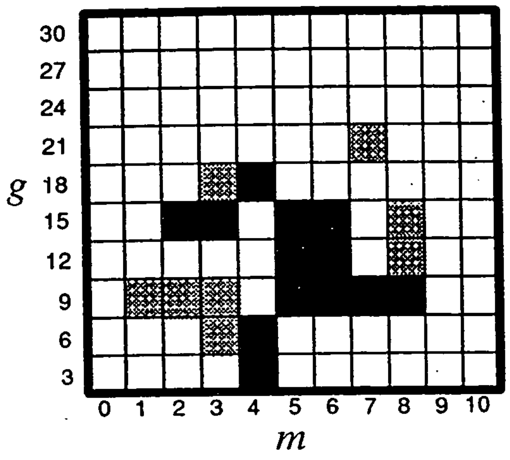
Background

Alignment has greatest power when applied to a family of related sequences, particularly, if structural data are available. With such data, the method becomes less sensitive to both the choice of gap penalty and amino acid relatedness matrix (the two weaknesses of the method, whatever the alignment algorithm). With good multiple sequence data most practical methods develop a consensus for each position in the alignment, giving, effectively, a specific local relatedness matrix. As for the case of gaps, as the alignment grows (aligning the most similar sequences first), early gaps (inserted with greatest confidence) provide preferred sites for further insertion.

Gap insertion has been thoroughly investigated.[16–19,21] However, these methods have been concerned mainly with the source of the local bias (typically, based on hydrophobicity or known structure) and not its implementation in the alignment algorithm. In the basic DP algorithm, the gap penalty is modified locally depending on known (or predicted) structure,

[21] R. F. Smith and T. F. Smith, *Protein Eng.* 5, 35 (1992).

a



b

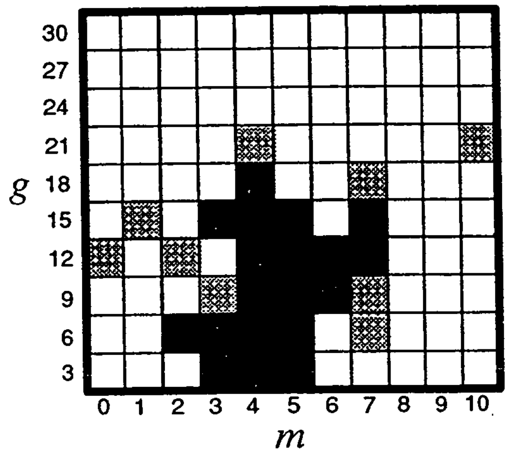


FIG. 4. Parameter-space plots of the acid protease family. The correct multiple alignment of the acid protease motifs is shown as filled cells for combinations of the gap penalty (g) and the matrix softness (m) parameters. The shaded cells indicate almost correct alignments in which only one motif in one sequence is misaligned. Plot (a) shows the results with the unmodified algorithm, whereas (b) shows the largest area found with the gap-bias algorithm using the parameter combination $t = 11$, $c = 5$ (see Table IV).

but this ignores the nature of the inserted sequence and also the opposing (broken) sequence ends. However, in an extended algorithm for structure comparison, Zhu *et al.*[22] combine these components (using observed solvent accessibilities), and, in the method described in this section, their approach is adopted but accessibility is substituted by a propensity of the sequence to be unstructured.

[22] Z.-Y. Zhu, A. Šali, and T. L. Blundell, *Protein Eng.* 5, 43 (1992).

Algorithm

Variable Gap Penalty Function. Aligning residue i with j and $i - k + 1$ with $j - 1$, the gap penalty function of Zhu *et al.*[22] can be written

$$s = k(p_j + p_{j-1} + v) + \sum_{m=1}^{k} q_{i-m} \tag{12}$$

where p is the propensity to be broken and q the propensity to be inserted, summed over k residues (v is a constant). Note that Eq. (12) is linearly dependent on length insert; this feature was not adopted, and instead a length-independent function was developed.

As predicted gap sites are less reliable than solvent exposure measurements, a range ($n = 3$) around the broken ends was considered, giving the three sums:

$$a_i = \sum_{m=1}^{k} q_{i-m}/k \tag{13}$$

$$b_j = \sum_{m=0}^{n-1} p_{j+m}/(m + 1) \tag{14}$$

$$c_j = \sum_{m=1}^{n} p_{j-m}/m \tag{15}$$

In addition, the flanking sums (b, c) were weighted by their distance from the break. The function allows the inserted segment and the flanking ends to have different propensities (p and q); however, little justification was seen for this, and a common propensity (p) was used (see below).

A score for insertion (t) can be formed as a function of the three sums (a, b, c) as

$$t = u[da + e(bc)^{1/2}]^f \tag{16}$$

with d, e, and u being weights. A product of ends (bc) was taken to balance the contributions on either side of the break and the square root then taken to moderate the resulting large contribution. Values of the parameter $f = 1$ and $f = \frac{1}{2}$ were investigated. When $f = 1$, $u = 0.01$; when $f = \frac{1}{2}$, $u = 0.1$. For use as a penalty, the function must be inverted, giving

$$r = g/(1 + t) \tag{17}$$

In this form the factor g is equivalent to the conventional gap penalty, which becomes reduced by t when it is favorable to insert a gap.

Empirical and Predicted Gap Propensities

Conservation and Variation. Sequence variation can be measured either by similarity, using one of the many relatedness matrices,[23–25] or by difference. The latter is less well quantified (except using simple identity), but the two approaches can usually be interconverted with little information loss.[26]

Given a matrix of differences (**D**), a score for sequence variation at a given position (k) is

$$v_k = \frac{2}{N(N-1)} \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} D_{a,b} \tag{18}$$

where a,b are indices of **D** for residues found in sequences i,j, respectively, in N aligned sequences. The parameter v was taken directly as the propensity p for insertion. Similarly, using a similarity matrix **M**,

$$w_k = \frac{2}{N(N+1)} \sum_{i=1}^{N} \sum_{j=i}^{N} M_{a,b} \tag{19}$$

and the inverted measure $100/w$ was taken as the propensity p.

Gaps do not appear in any of the above matrices and so must be treated separately. Relative to a matched identity score of 10, an acid/gap match scored $10 + h$ and a gap/gap match scored $10 + 2h$. When using a similarity matrix, gaps were taken to have no similarity to any acid or themselves: their effect was therefore dependent on the absolute values in the relatedness matrix.

Pascarella and Argos Scales. Preference parameters for the likelihood of gap insertion have been calculated by Pascarella and Argos[27] from their database of alignments based on known structures. Ideally for the current application, the preferences were calculated separately for the inserted region (subdivided by size) and the residues on the broken ends of the other sequence. The results of this analysis are summarized in Table V by ranking the amino acids according to their propensity.

The preference scales of Pascarella and Argos[27] span a range that is roughly an order of magnitude larger than their mean standard deviation. While this is sufficient for all the scales to exhibit the expected rough partition of hydrophobic and hydrophilic properties, there is little significant

[23] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 345. National Biomedical Research Foundation. Washington, D.C., 1978.

[24] S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 10915 (1992).

[25] D. T. Jones, W. R. Taylor, and J. M. Thornton, *CABIOS* **8,** 275 (1992).

[26] W. R. Taylor and D. T. Jones, *J. Theor. Biol.* **164,** 65 (1993).

[27] S. Pascarella and P. Argos, *J. Mol. Biol.* **224,** 461 (1992).

TABLE V
RANKED PASCARELLA–ARGOS PREFERENCES[a]

	Inserts			
Ends	1–5	6–10	11–15	16–56
G	N	S	T	W
N	S	G	P	Y
R	G	T	N	G
P	P	Q	Y	E
S	T	D	D	H
T	D	N	Q	R
K	Q	P	E	L
D	K	A	G	N
H	H	F	K	S
Y	A	R	S	D
Q	E	Y	V	T
A	L	C	H	K
C	R	K	L	Q
F	Y	V	M	A
L	F	L	I	I
W	V	W	A	P
V	W	E	R	M
M	M	H	F	F
E	I	I	C	V
I	C	M	W	C

[a] The preferences calculated by Pascarella and Argos[27] have been simplified to a rank ordering. These were calculated for the residue at the broken ends of the insert and the residues found in the insert (classed by the length ranges shown). In the ends and shorter inserts there is a clear preference for G, P, and the small hydrophilic amino acids (D, N, S, and T).

separation between neighbors in the rank order of the acids (Table V), and much of the variation between scales can be interpreted as sampling error, especially for the less numerous larger sized insertions. Within this noise level, there is also little to distinguish the preferences within insertions from those for the broken ends. This can be seen more graphically by combining the results for the insertions as a weighted mean and plotting these values against the scale for the broken ends (Fig. 5).

The broken-end scale of Pascarella and Argos[27] has been used to good effect by Thompson *et al.,*[15] and although it would be possible to combine this with the values for the inserted segment,[27] they used instead a simple

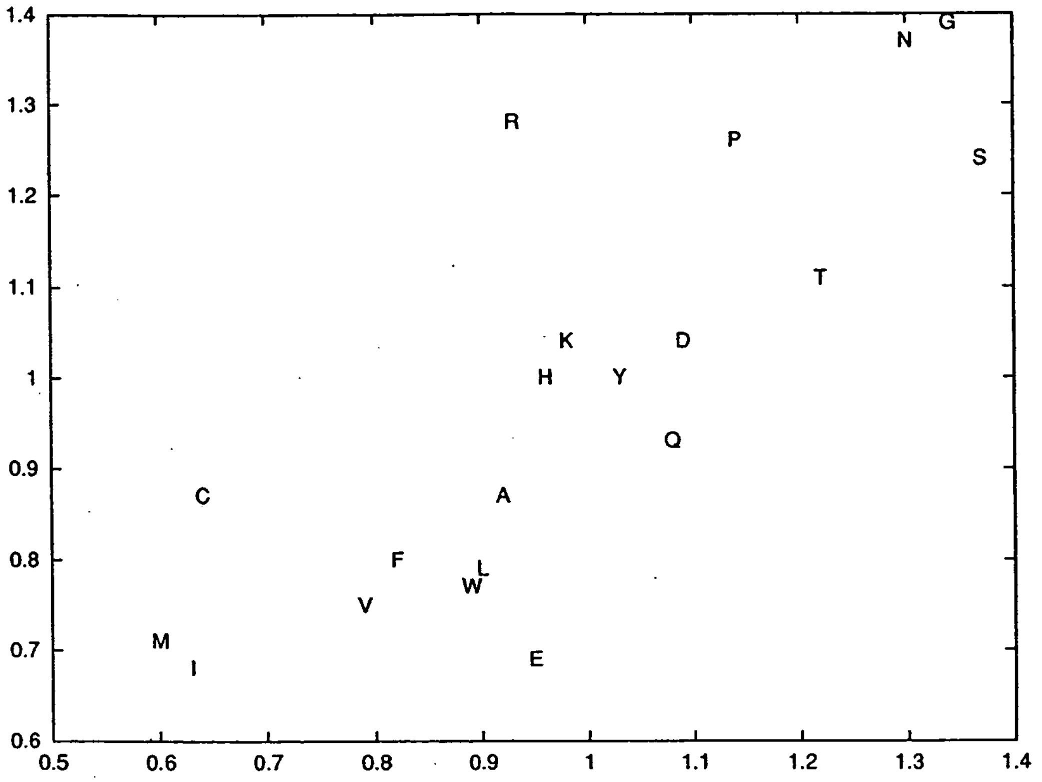


FIG. 5. Comparison of Pascarella and Argos end and insert preferences. The preferences of Pascarella and Argos[27] for a residue to be adjacent to a gap are plotted against the weighted mean of the preferences to be in an insert of any size. The average error in the y axis (ends) is 0.06, and that on the x axis (inserts) is roughly twice this value. Therefore, only arginine (R), cysteine (C), and glutamate (E) deviate significantly.

hydrophilic bias for insertion. It seems doubtful whether there is sufficient distinction among the scales to justify their full implementation.

Chou and Fasman Preferences. A simpler source of preferences can be found in the propensities for residues to be in secondary structure. This source is attractive as it avoids the inherent circularity of deriving parameters to control alignments, from alignments. The simplest source of preferences was taken from the method of Chou and Fasman[28] rather than the more complex methods, such as that of Garnier *et al.*,[29] which consider a local region of sequence.

Concentrating on gaps, the type of secondary structure is not important, only whether there is secondary structure present. For this two-state prediction, preference parameters were not recalculated, but instead a simple

[28] P. Y. Chou and G. D. Fasman, *Adv. Enzymol.* **47,** 45 (1978).
[29] J. Garnier, D. J. Osguthorpe, and B. Robson, *J. Mol. Biol.* **120,** 97 (1978).

score (p) was based on the propensities for turn (P_t), α helix (P_α) and β sheet (P_β) as

$$p = P_t/(P_\alpha + P_\beta) \tag{20}$$

The order of the amino acids ranked on p (Table VI), conforms to expectations: those with unique main-chain stereochemistry (G, P) are highest, followed by those with small, polar side chains (S, N, D), with all the hydrophobic amino acids at the bottom (A, W, F, L, I, M, V). In absolute terms, the scale expands at the hydrophilic end; however, if the logarithm of the values is taken, the scale becomes comparable to those of Pascarella and Argos[27] (Fig. 6). The outlying points on this plot correspond to arginine (R), cysteine (C), proline (P), and glutamate (E), with

TABLE VI
SECONDARY STRUCTURE-BASED GAP PREFERENCE SCALES[a]

Amino acid	P_α	P_β	P_t	$p - p_{min}$	ln p
P	0.57	0.55	1.59	11.95	2.65
G	0.57	0.75	1.50	9.12	2.43
S	0.77	0.75	1.32	6.44	2.16
N	0.67	0.89	1.35	6.41	2.16
D	1.01	0.54	1.20	5.50	2.05
C	0.70	1.19	1.18	4.00	1.83
H	1.00	0.87	1.06	3.42	1.73
R	0.98	0.93	1.04	3.20	1.69
T	0.83	1.19	1.07	3.05	1.67
K	1.16	0.74	0.98	2.91	1.64
Y	0.69	1.47	1.06	2.66	1.59
E	1.51	0.37	0.84	2.22	1.50
Q	1.11	1.10	0.86	1.65	1.36
A	1.42	0.83	0.70	0.86	1.13
W	1.08	1.37	0.75	0.81	1.12
F	1.13	1.38	0.71	0.58	1.04
L	1.21	1.30	0.68	0.46	1.00
I	1.08	1.60	0.66	0.22	0.90
M	1.45	1.05	0.58	0.07	0.84
V	1.06	1.70	0.62	0.00	0.81

[a] The Chou and Fasman[28] propensities for amino acids to be in α, β, and turn are tabulated with the derivative measure p and its logarithm (on which the entries are ranked). WIth few exceptions this latter measure corresponds well with the Pascarella and Argos[27] scales.

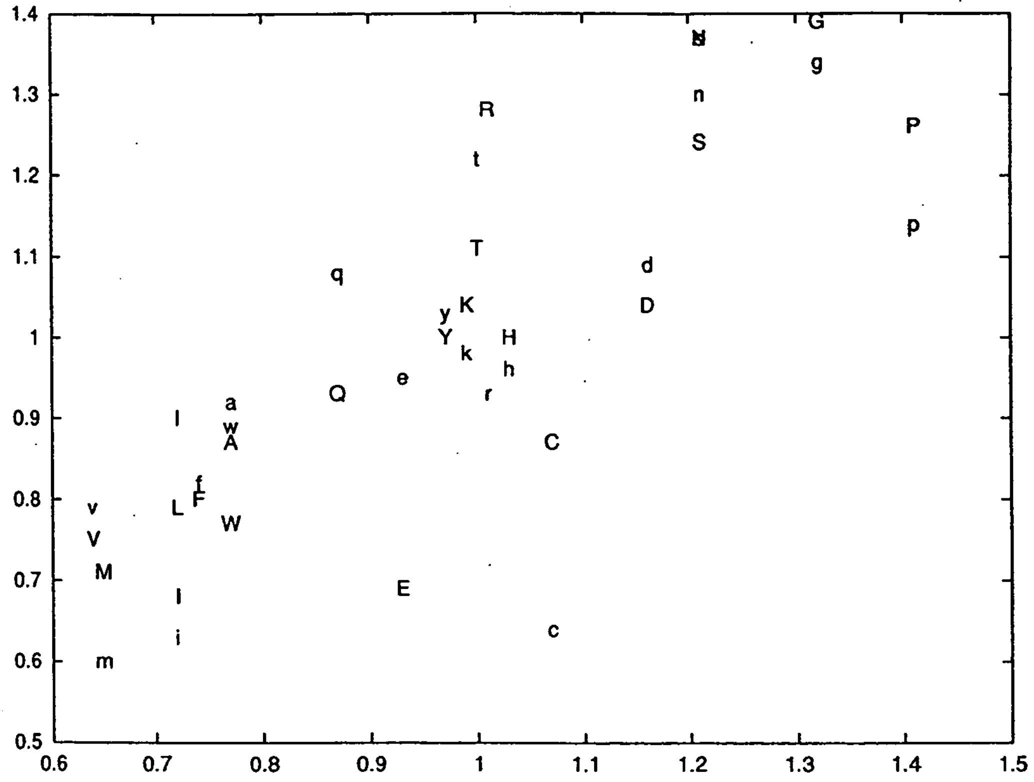


FIG. 6. Comparison of Chou–Fasman and Pascarella–Argos preferences. The preferences of Pascarella and Argos[27] for a residue to be associated with a gap (Fig. 5) are plotted on the y axis against the scale (p) derived from the Chou–Fasman propensities for secondary structure (Table VI). The linear scaling of the p values was found that gave the minimum sum-of-squares deviation from both Pascarella–Argos scales (ends and inserts). Points are plotted using the one-letter amino acid code in uppercase for the Pascarella–Argos ends values and lowercase for the Pascarella–Argos insert values.

the three latter acids having a greater propensity to associate with gaps by the p measure.

Application

Calculation of Phase Overlap. As in the previous studies described above, the results of the algorithm were assessed by the size of the phase corresponding to the correct alignment in the parameter space. The parameter space of the two weights $\{d,e\}$ (corresponding to the inserted segment and the flanking ends) was investigated over a square at 132 points from $d = e = 0$ to $d = e = 10$. For each point the area of the correct alignment in the parameter space defined by gap penalty g and matrix softness m was found.

Areas were found for two protein families and their overlap expressed as the percentage size of the union of the two areas. (Perfect overlap would thus score 100 and no overlap 0.) This comparative use of two proteins greatly reduces the possibility of achieving trivial improvements simply by the indirect rescaling of one of the parameters defining the observation space and its adoption as a generally robust protocol will be discussed further below.

The globins and aspartyl proteases were again employed as test data.

Unbiased Method. Without any modification, the basic DP algorithm, as implemented in MULTAL,[7,8] did not result in any overlap of the correct alignment phase for the two proteins. All the overlaps described below are therefore an improvement over the basic algorithm.

Conservation-Based Measures. Using the Dayhoff PAM120 matrix and $f = 1$ [Eq. (16)] produced a band of slight improvement in overlap running antidiagonal across the matrix; similarly, with the square-root variation [$f = \frac{1}{2}$, Eq. (16)], an equivalent stripe across the matrix was observed (Table VII).

Difference-Based Measures

No special gap treatment. The difference measure based only on amino acid identity (with $f = 1$) gave substantial increases in overlap over a broad range of weight combinations, with the best results obtained when both

TABLE VII
PERCENTAGE PHASE-AREA OVERLAP WITH AMINO ACID SIMILARITY ($f = \frac{1}{2}$)[a]

	d										
e	0	1	2	3	4	5	6	7	8	9	10
0								10	10	9	9
1						9	20	10	8	10	8
2						8	11	10	9	10	8
3					8	20	11	9	11	8	9
4			13		8	11	11	10	11	8	8
5			10		10	11	11	14	8	8	8
6		10		8	11	11	10	11	8	8	
7		10	10	20	11	10	11	10			
8	13		9	9	11	10					
9	13	10	20	10							
10	13	9									

[a] Only nonzero values are entered. The weights *d* and *e* were applied to the inserted component and the broken-end components.

TABLE VIII
PERCENTAGE PHASE-AREA OVERLAP WITH AMINO ACID DIFFERENCES ($f = \frac{1}{2}$)[a]

	d										
e	0	1	2	3	4	5	6	7	8	9	10
0		14					4				
1		18	21	14	18	8	10	3	7	7	13
2		11	26	21	21	15	9	3	15	6	12
3		6	9	16	15	15	12	11	10	6	5
4		14	5	12	14	11	14	21	17	11	9
5				3	15	14	14	17	13	10	10
6				4	17	6	18	17	14	9	16
7		15	5	9		8	18	14	12	11	10
8			11			12	19	14	6	9	8
9		12	4	4			6	10	14	12	12
10		13	5			7	3	8	13	14	14

[a] Only nonzero values are entered. The parameters *d* and *e* are the insert and end weights.

components were weighted. This synergism between the components was even more marked with $f = \frac{1}{2}$ (Table VIII).

Special gap treatment. Setting the gap bonus parameter $h = 5$ with $f = \frac{1}{2}$, made little difference, giving similar results to the unweighted gaps (Table VIII) and exhibiting a wedge of good improvement expanding from the origin (Table IX). Increasing the gap bonus weight to $h = 10$ produced only slight improvement around the diagonal.

Analysis of Results. Cooperativity between the two components (broken ends and insert) was found only with the difference-based measure. This was unexpected and is difficult to rationalize but may be associated with the different softness of the similarity and difference matrices. A position conserving, say, I, L, V, would appear conserved by amino similarity but appear divergent based on a count of different identities. Further studies will be necessary to see if this is a general phenomenon.

Conclusions

Theoretical Considerations

Strictly, the dynamic programming algorithm should be applied only with a metric scoring function; that is, the optimal path can be cut anywhere and the resulting fragments are themselves optimal alignments. The algo-

TABLE IX
PERCENTAGE PHASE-AREA OVERLAP WITH AMINO ACID DIFFERENCES ($f = \frac{1}{2}$) AND FURTHER ENHANCED GAP SIGNIFICANCE ($h = 5$)[a]

	d										
e	0	1	2	3	4	5	6	7	8	9	10
0		20									
1		28	26	19	16						10
2			22	25	19	17	17		10	10	10
3			14	24	19	19	15	13	16	10	11
4		13		15	18	16	20	24	20	14	13
5					21	13	19	17	15	14	14
6		13			18		23	24	23	16	19
7						13	20	17	16	16	17
8						13	18	13	13	12	12
9		13						12	19	18	15
10		10					10	12	15	14	15

[a] Only values over 9 are entered. The parameters *d* and *e* are the insert and end weights.

rithms presented in this work, to varying degrees, violate that condition as the maximal path is determined at one end of the sequence by considering information obtained directly from the other end (and not through a locally accumulating sum). The alignment at one end of the sequences (or one end of a gap) thus has a view of the sequence that was not available when the other end was aligned, so breaking the symmetry of the calculation. This implies that if the algorithm were applied to the sequences in reverse, the same answer (alignment and score) might not be obtained. The very existence of such a possibility means that the algorithm cannot guarantee to find the optimal solution.

The related problems of nonoptimality and directional asymmetry would normally be seen as good reasons to abandon the DP algorithm in favor of an algorithm that can guarantee the optimum soluton. However, problems of the type reviewed in the current work (that consider nonlocal interactions) correspond in complexity to the related problems of structure alignment[30,31] and optimal sequence threading,[32] which have been shown to be NP-complete in computational complexity.[33] The only algorithm then available to guarantee an optimal solution is combinatoric enumeration or,

[30] W. R. Taylor and C. A. Orengo, *J. Mol. Biol.* **208**, 1 (1989).
[31] A. Šali and T. L. Blundell, *J. Mol. Biol.* **212**, 403 (1990).
[32] D. T. Jones, W. R. Taylor, and J. M. Thornton, *Nature* (*London*) **358**, 86 (1992).
[33] R. H. Lathrop, *Protein Eng.* **7**, 1095 (1994).

with less certainty, stochastic searching (e.g., simulated annealing). As both these approaches are computationally very expensive on sequences of sufficient length to be of interest, there is strong motivation to retain the dynamic programming approach and either quantify the probable error sizes[2] or deal with them by iteration.[34]

Assessing Alignment Quality

A general problem that has recurred throughout the preceding descriptions is the measurement of alignment accuracy. With known structures, alignment can be based on structural geometry,[30] giving a criterion of truth against which pure sequence alignment can be compared and assessed. With remotely related sequences however, comparison even with known structures can become ambiguous or systematically differ from expectation. For example: the structural alignment of a β hairpin in two proteins might give

```
BBBBB---**---BBBBB
BBBBBBBB**BBBBBBBB
```

where B is a residue in a β strand and * indicates equivalent positions in the turn (– is a gap). Unless the turn is strongly conserved in the sequence, it is unlikely that this alignment would be reproduced with only sequence data: more probable would be the insertion of a single gap.

A measure of alignment quality is needed for remotely related sequences that does not depend on the unreliable exact location of gaps. Previously, motifs have been used to circumvent this problem,[35,36] and, while effective, this approach brings its own problem in the choice of motifs to monitor. If these are too few they will not give a full reflection of alignment quality, but if too many are used, the problems associated with single residue matching reemerge. A balance has been found in the work described above by using relatively few motifs (between two and ten) associated with well-spaced core secondary structures.

The use of relatively few motifs, however, can result in the situation where all motifs are easily aligned, creating a lack of discriminating power (if two parameter sets both get the motifs aligned correctly, it is not possible to decide which is better). This problem was overcome through monitoring the area in parameter space corresponding to the correct alignment. This approach is particularly useful as it quantifies the stability of the correct

[34] C. A. Orengo and W. R. Taylor, *J. Theor. Biol.* **147,** 517 (1990).

[35] W. R. Taylor, *J. Mol. Biol.* **188,** 233 (1986).

[36] M. A. McClure, T. K. Vasi, and W. M. Fitch, *Mol. Biol. Evol.* **11,** 571 (1994).

alignment allowing parameter sets to be discriminated by their distance from any incorrect alignment.

Unfortunately, the use of phase areas in parameter space to monitor performance leads to the complication that the area might be trivially changed by modifying a parameter that correlates with a dimension of the space being monitored. For example; if one dimension was the gap penalty, then a new method that simply halved the gap penalty would increase the area of the correct alignment phase by 50%. In the first study described above, where such correlation was possible, it was monitored by taking the number of gaps as an invariant control.[1]

Although the number of gaps is satisfactory, it is not an ideal control, and it was avoided in the later study of structure-biased gap penalties where the coupling between area and parameter is direct. In this work, a better control was found by simultaneously monitoring the phase space of two protein families and using the percentage overlap as a measure of improvement. This is exactly the requirement of practical studies, such as multiple alignment or databank scanning, in which an investigator takes a default parameter set and expects it to be optimal over as wide a range of proteins as possible.

Having established the protocol many further applications await. Of great practical importance would be to search for the parameter combination that gives the correct alignment for the maximum number of sequence families in the data banks. In a more theoretical direction, the phase space of the correct alignments could be investigated at increasing sample densities to delineate their outlines. With just two sequences these areas are polygons,[37] but given the much greater complexity of multiple alignment, irregular (possibly fractal) shapes would be expected.

Integration with Iteration

The algorithms described above have been coded into separate modified versions of MULTAL.[7,8] Each method still requires considerable work to fully explore their individual behavior over a larger number of proteins, however, and work is currently underway to combine the methods into a common implementation that uses iteration to overcome directional asymmetry in the calculations and simultaneously provide a stopping condition.

[37] M. Vingron and M. S. Waterman, *J. Mol. Biol.* **235**, 1 (1994).

[21] Progressive Alignment of Amino Acid Sequences and Construction of Phylogenetic Trees from Them

By DA-FEI FENG and RUSSELL F. DOOLITTLE

Introduction

In 1970, Needleman and Wunsch published an elegant algorithm for optimally aligning two protein sequences.[1] Moreover, the scheme could be used with weighting scales that assigned scores to each set of paired residues and assessed penalties for unpaired ones (gaps). Arguably, the most popular of these amino acid substitution matrices was the PAM scale devised by Dayhoff and co-workers, which was based on observed mutations for sets of closely related protein sequences.[2,3] During the 1970s and early 1980s, numerous investigators used the Needleman–Wunsch algorithm in conjunction with various versions of the Dayhoff PAM matrix to generate binary (pairwise) alignments. On the other hand, multisequence alignments were usually made manually, the binary alignments serving as a guide.

Although in principle the Needleman and Wunsch approach could be extended to multiple sequences, in practice the computational time for such a task proved prohibitively long, the memory required for storing the necessary arrays being enormous even for sets of very short sequences.[4,5] Although global methods were devised that yielded reasonable alignments on longer sequences, they were still restricted to five or fewer sequences.[6] An even more disappointing aspect was that the pattern of gaps in multiple alignments was often inconsistent with what was observed in binary alignments; thus, the optimal alignment between the two closest sequences as indicated by a binary alignment was often altered in the presence of a third or fourth sequence.[6]

[1] S. B. Needleman and C. D. Wunsch, *J. Mol. Biol.* **48,** 443 (1970).

[2] M. O. Dayhoff, R. V. Eck, and C. M. Park, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, p 89. National Biomedical Research Foundation, Washington, D.C., 1972.

[3] R. M. Schwartz and M. O. Dayhoff, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 353. National Biomedical Research Foundation, Washington, D.C., 1978.

[4] R. A. Jue, N. W. Woodbury, and R. F. Doolittle, *J. Mol. Evol.* **15,** 129 (1979).

[5] M. Murata, J. S. Richardson, and J. L. Sussman, *Proc. Natl. Acad. Sci. U.S.A.* **82,** 3073 (1985).

[6] M. S. Johnson and R. F. Doolittle, *J. Mol. Evol.* **23,** 267 (1986).

In 1987, we reported a simple progressive alignment procedure that circumvented the need for aligning sequences exhaustively.[7] Like other methods introduced at about the same time,[8,9] it actually used a binary alignment algorithm[1] iteratively, first to align the two most closely related sequences and then to align the next most similar one to those, etc. The strategy was further guided by the simple rule "once a gap, always a gap," the rationale being that the positions and lengths of gaps introduced between the more similar pairs of sequences should not be affected by distantly related ones.[7] As such, sequences were compared in an approximately descending order of similarity, each new overall alignment score being determined by comparisons of residues of the last added sequence against the averaged scores of all previously aligned residues. The program was written in such a way that the order could be refined automatically by checking the next two sequences at each round, priority being determined on the basis of the higher similarity score. This operation was continued until all the sequences were aligned. We referred to this procedure as progressive alignment. Because progressive alignment is not exhaustive, the required computational time is short compared with global alignment procedures.

Progressive Alignment and Phylogenetic Trees

Although many investigators are only interested in alignments per se, our interest has been driven by a need for the automatic construction of sequence-based phylogenetic trees. Historically, the first step in constructing a quantitative phylogenetic tree has always been to find an objective procedure for obtaining a multiple alignment. In our 1987 paper[7] we presented a suite of programs that allowed users to go directly from the aligned sequences to a phylogenetic tree, a number of different programs being used sequentially. Some steps had to be performed by hand, however, and we later described a version in which many of the steps were tied together in a more user-friendly fashion.[10] Even then, the system still had some annoying limitations, however. In this chapter we describe a much improved, easier to use, set of programs, which we refer to as ProPack: a packet of programs centering around progressive alignment (Table I). The changes pertain mainly to speeding up the calculations and reducing the number of manual operations; the basic idea of progressive alignment remains the same.

[7] D. F. Feng and R. F. Doolittle, *J. Mol. Evol.* **25,** 351 (1987).
[8] W. R. Taylor, *CABIOS* **3,** 81 (1987).
[9] G. J. Barton and M. J. E. Sternberg, *J. Mol. Biol.* **198,** 327 (1987).
[10] D. F. Feng and R. F. Doolittle, this series, Vol. 183, p. 375.

TABLE I

ProPack Suite of Programs for Progressive Alignment and Making Trees

Program	Objective
FORMAT	Convert sequences to Old Atlas format
INSPECT	Find matching boundaries
CROP	Cut sequences to order
ARRANGE	Determine an approximate branching order
PREALIGN	Prealign cluster of sequences
ALIGN	Determine multiple alignment only
MULPUB	Change aligned sequences to compact format
TREE	Make a multiple alignment and construct a phylogenetic tree from it
BLEN	Determine branch lengths based on distance matrix
NONEG	Find best branching order with no negative branch lengths
SETREE	Convert information to a format recognized by tree drawing program
BTREE	Sample multiple alignments and generate new trees for bootstrapping procedure
CLUS	Analyze clusters (nodes) from BTREE for agreement with initial tree

Improvements

The principal new features are (a) introduction of an option for choosing an alternative amino acid substitution matrix, (b) automatic elimination of negative branch lengths in the calculation of phylogenetic trees, (c) addition of a bootstrap analysis option, and (d) inclusion of a simple program for converting output to a form that can be used to draw trees on a microcomputer with standard software. Additionally, experience has shown that the prealignment of subclusters is seldom necessary, and as a result the program PREALIGN is downplayed in the new ensemble. Instead, sequences can be input in any order, with the program ARRANGE automatically sorting them into an approximate order for processing by the main program TREE. A summary of these programs is presented in Table I, and a flowchart of the new scheme is depicted in Fig. 1.

Programs

The computer programs described here are for protein sequence comparisons (not DNA or RNA). They are written in the C language[11] and run under the UNIX operating environment, although they have also been used successfully by others with a VMS system after minor modifications.

[11] B. W. Kernighan and D. M. Ritchie, "The C Programming Language." Prentice-Hall, Englewood Cliffs, New Jersey, 1978.

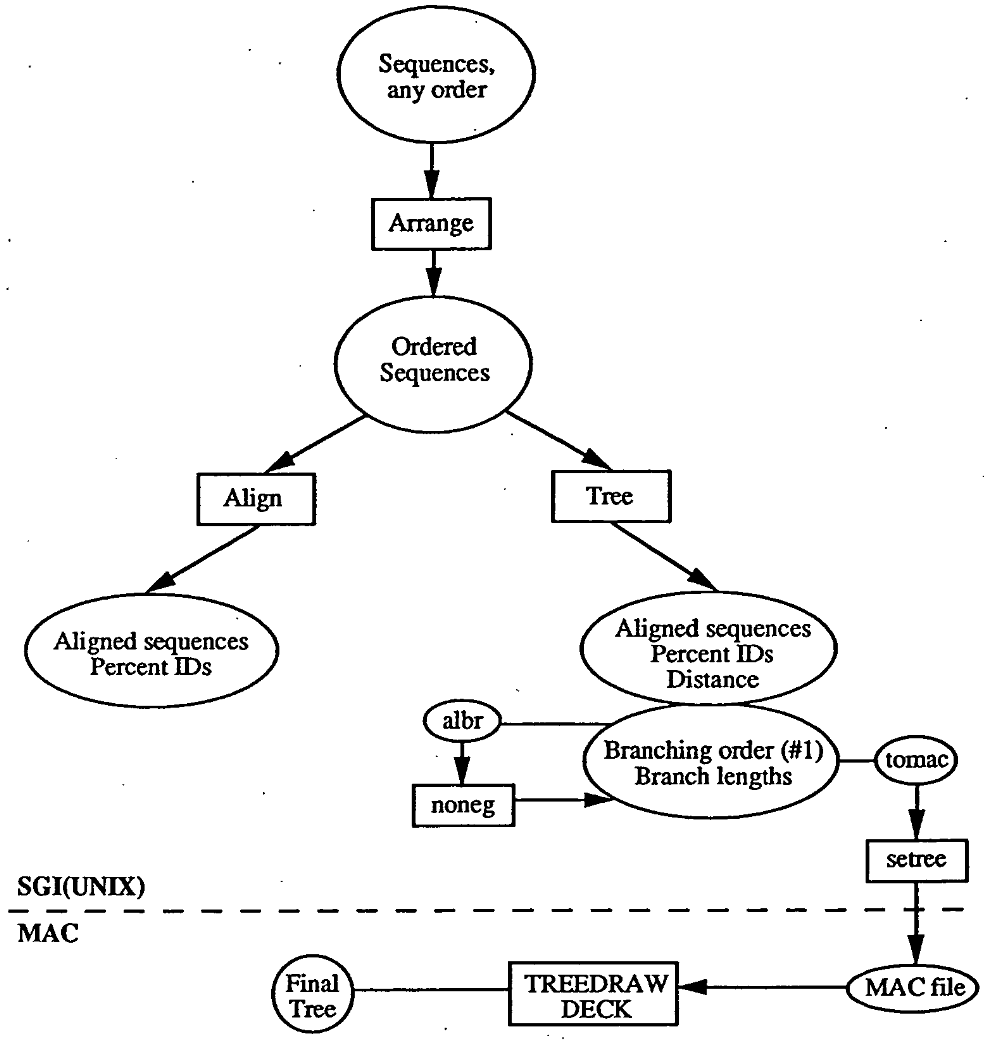


FIG. 1. Flow diagram of progressive alignment and phylogenetic tree construction. The names in ovals denote files; square boxes represent programs. All programs shown above the dashed line run under the UNIX operating system via a command line. A description of the command line format for each of the programs can be obtained simply by typing the name of the program after these programs have been compiled on the home computer.

In their present form they are exactly adapted for running on Silicon Graphics machines (IRIX 4.0.5 IOP).

The primary program TREE, which produces the multiple alignment, the branching order, and the branch lengths, can handle as many as 52 sequences (a number that is incidental to the number of lowercase and uppercase letters that are used as designation for entries). The maximum lengths of the sequences are strictly a function of the available computer memory. At the present time, sequences of 1000 residues can be aligned easily. The sequences must be provided in the "Old Atlas" format (Fig.

```
TDEHU L-lactate dehydrogenase (EC 1.1.1.27) chain M - human
PDEHU     1 K I T V V G V G A V G M A C A I S I L M K D L A D E L A L V
PDEHU    31 D V I E D K L K G E M M D L Q H G S L F L R T P K I V S G K
PDEHU    61 D Y N V T A N S K L V I I T A G A R Q Q E G E S R L N L V Q
PDEHU    91 R N V N I F K F I I P N V V K Y S P N C K L L I V S N P V D
PDEHU   121 I L T Y V A W K I S G F P K N R V I G S G C N L D S A R F R
PDEHU   151 Y L M G E R L G V H P L S C H G W V L G E H G D S S V P V W
PDEHU   181 S G M N V A G V S L K T L H P D L G T D K D K E Q W K E V H
PDEHU   211 K Q V V E S A Y E V I K L K G Y T S W A I G L S V A D L A E
PDEHU   241 S I M K N L R R V H P V S T M I K G L Y G I K D D V F L S V
PDEHU   271 P C I L G Q N G I S D L V K V T L T S E E E A R L K K S A D
PDEHU   301 T L W G I Q *
```

FIG. 2. Example of Old Atlas format.

2). The program FORMAT changes sequences into the proper form so that they are recognized by all the programs.

Although program names are capitalized throughout this chapter in order to emphasize what is actually typed, on a day-to-day basis we ordinarily use lowercase. It must be emphasized that UNIX is rigorously case sensitive.

Converting Similarity to Distance

Similarity scores for pairs of aligned sequences are converted to difference scores by the following version of the Poisson equation:[12]

$$D = -\ln S \tag{1}$$

where

$$S = [S_{\text{real}}(ij) - S_{\text{rand}}(ij)]/[S_{\text{iden}}(ij) - S_{\text{rand}}(ij)] \times 100 \tag{2}$$

$S_{\text{real}}(ij)$ is the observed similarity score for the two sequences being aligned, and $S_{\text{iden}}(ij)$ is the average of the two scores for the two sequences compared with themselves. $S_{\text{rand}}(ij)$ provides a measure of the background noise between sequences *i* and *j*. Previously $S_{\text{rand}}(ij)$ was determined by a computer-intensive shuffling operation; now it is calculated by a formula, as discussed below.

S_{rand}

The random score between two optimally aligned sequences, $S_{\text{rand}}(ij)$, is a function of the composition of the sequences, the number of internal gaps in the multiple alignment, the gap penalty, and the scoring matrix used. To put this on a mathematical basis, the following conditions obtained:

[12] D. F. Feng, M. S. Johnson, and R. F. Doolittle, *J. Mol. Evol.* **21,** 112 (1985).

a_i, b_j represent the residue type in sequence i and j, $N_a(i)$, $N_b(j)$ are the number of times a_i, b_j appear in sequences i and j, N_g is the number of internal gaps unique to either sequence i or j, irrespective of their lengths, *pen* is the gap penalty used in the calculation, and L is the overall length of the sequences after they are aligned. Now, if we avail ourselves of a suitable amino acid substitution matrix M, then

$$S_{\text{rand}} = (1/L) \Sigma \Sigma M(a_i, a_j)N_a(i)N_b(j) - N_g * pen \qquad (3)$$

Equation (3) allows S_{rand} to be calculated independently of a random number generator; not only is the process speeded up, but it is more consistent.

Outline of Operations

In essence, there are 10 steps to making a sequence-based tree with our programs: (a) gather the sequences, (b) format and otherwise edit the sequences, (c) catenate the sequences into a single file, (d) arrange the sequences into an appropriate order, (e) align the sequences, (f) calculate the similarity scores and pairwise distances, (g) make a distance (difference) matrix, (h) determine the branching order and branch lengths, (i) if necessary, find an alternate solution without negative branch lengths, and (j) draw a tree. One also has the option of performing a bootstrap analysis, and we comment further on this subsequently.

Input

Gathering the sequences from appropriate databases is a straightforward matter. To put them into a suitable format for our programs, one must first strip out all nonsequence alphabetic characters, including the title (numbers will be ignored).

```
FORMAT temp >seq1
```

The program will then ask for an appropriate title and a four-character identifier. Once a sequence is in the "old Atlas" format, it can also be readily edited with a program called CROP so that only specified segments are called on.

```
CROP seq1 23 487 >seq1X
```

In this case an edited sequence composed of residues 23 to 487 is put into a separate file. CROP is a very useful program because good alignments of distantly related sequences cannot be obtained if the lengths of the sequences are too different. In this regard, the program INSPECT[13] is

[13] R. F. Doolittle, "URFs and ORFs: A Primer on How to Analyze Derived Amino Acid Sequences," p. 21. University Science Books, Mill Valley, California, 1986.

helpful for identifying appropriate cut points in sequences of different lengths or mosaic composition.

Once the sequences have been gathered, formatted, and cropped to specification, they need to be catenated into a single file. As suggested above, the order is important. Often it is sufficient to input the sequences in an approximate biological order. A more rigorous way is to find an approximate branching order based on pairwise alignments only. The program called ARRANGE makes an alignment of every pair and uses the scores to find an approximate branching order by the method of Fitch–Margoliash.[14] This approximate branching order can then be used as a guide to catenating the sequences into a starting file for the multiple alignment.

Alignment Strategy

Once the sequences are in a single file in an approximate order, they can be aligned with a program called ALIGN. Consider what goes on in this program. Let A, B, C, D, E, ..., be a set of sequences in descending similarity order as determined by ARRANGE. The ALIGN program first aligns A and B. If (AB) represents an optimal alignment between these two sequences, then C(AB) and (AB)C are tried next, the set with the higher similarity score determining whether C(AB) or (AB)C will be the three-way alignment. Assuming that (AB)C has a higher score, the next step involves trying the two alignments, ((AB)C)D and ((AB)D)C. Again, the higher similarity score determines whether ((AB)C)D or ((AB)D)C will be the four-way alignment. This procedure is repeated until all the sequences are aligned. Because of this iterative process, the order of the sequences in the resultfile alignment may be different from that generated by ARRANGE as presented in seqfile.

ALIGN seqfile seqfile1 M > resultfile

where M is the amino acid mutation matrix which will be discussed in more detail below. Seqfile1 is a file written during execution that contains the sequences with X's inserted as neutral elements at all gap positions. These elements are neutral in the sense that the matching of an X with a residue in the other sequence results in a value of zero. The number of X's tends to increase as more sequences are brought into the alignment, the direct result of the rule "once a gap, always a gap." In addition to its use in scoring within the ALIGN program itself, the file is also used as the input

[14] W. M. Fitch and E. Margoliash, *Science* **155,** 279 (1967).

file for MULPUB, a program that converts the arrangement to a close-pack form of any specified size (in column lengths). Thus,

```
MULPUB seqfile1 80 > seqfileM
```

rearranges the aligned sequences in a close-packed array 80 columns wide (Fig. 3).

Choice of Matrices

In addition to the Dayhoff (PAM250) matrix,[3] which has been used commonly in the past, a number of other scoring matrices have been published, including one by Gonnet *et al.*[15] and another based on block alignments by Henikoff and Henikoff.[16] Either of these matrices can be used with the ARRANGE, ALIGN, or TREE programs. In an extensive study involving enzyme sequences, we found that the differences between the PAM250 and the Gonnet matrices are small, and both produce reliable results for sequences that are closely related to each other. The BLOSUM62 matrix, on the other hand, appears to give better alignments for more distantly related sequences. For the PAM250 and Gonnet matrices the gap penalty should be set at 8; in the case of the BLOSUM62 matrix, we have found a gap penalty of 6 to be optimum. These values were initially determined by a consideration of the distribution of the scores used in these matrices and then verified empirically by inspection of the alignments produced by them. The Dayhoff PAM250, Gonnet, or BLOSUM matrices can be called simply by adding a D, G, or B to the command line, respectively.

```
ALIGN seqfile seqfile1 D > resultfile     (use Dayhoff PAM250, gap penalty = 8)
```

or

```
ALIGN seqfile seqfile1 G > resultfile     (use Gonnet, gap penalty = 8)
```

or

```
ALIGN seqfile seqfile1 B > resultfile     (use BLOSUM62, gap penalty = 6)
```

Making a Tree

Alternatively, the sequences can be aligned and a tree constructed directly by the program called TREE. The early operations of the ALIGN

[15] G. H. Gonnet, M. A. Cohen, and S. A. Benner, *Science* **256,** 1443 (1992).
[16] S. Henikoff and J. G. Henikoff, *Proteins: Struct. Funct. Genet.* **89,** 10915 (1992).

```
                  * * **            * *    *           *                               **  *
DEHU         KITVVGVGAVGMACAISILMKDLADELALVDVIEDKLKGEMMDLQHGSLFLRTPKIVSGKDYNVTANSKLVIITAGARQ
DECH         KISVVGVGAVGMACAISILMKDLADELTLVDVVEDKLKGEMMDLQHGSLFLKTPKITSGKDYSVTAHSKLVIVTAGARQ
DEDF         KITVVGVGAVGMACAISILMKDLADEVALVDVMEDKLKGEMMDLQHGSLFLHTAKIVSGKDYSVSAGSKLVVITAGARQ
A382        SKVTIVGVGQVGMAAAISVLLRDLADELALVDVVEDRLKGEMMDLLHGSLFLKTAKIVADKDYSVTAGSRLVVVTAGARQ
A360        TKISVIGAGNVGMAIAQTILTQNLADEIALVDALPDKLRGEALDLQHAAAFLPRVRI SGTDAAVTKNSDLVIVTAGARQ
S224        TKVSVIGAGNVGMAIAQTILTRDLADEIALVDAVPDKLRGEMLDLQHAAAFLPRTRLVSGTDMSVTRGSDLVIVTAGARQ
DELB         KVILVGDGAVGSSYAYAMVLQGIAQEIGIVDIFKDKTKGDAIDLSNALPFTSPKKIYSA EYSDAKDADLVVITAGAPQ
DEBS         RVVVIGAGFVGASYVFALMNQGIADEIVLIDANESKAIGDAMDFNHGKVFAPKPVDIWHGDYDDCRDADLVVICAGANQ
B408        QKVVLVGDGAVGSSYAFAMAQQGIAEEFVIVDVVKDRTKGDALDLEDAQAFTAPKKIYSG EYSDCKDADLVVITAGAPQ
S368        MKIGIVGLGRVGSSTAFALLMKGFAREMVLIDVDKKRAEGDALDLIHGTPFTRRANIYAG DYADLKGSDVVIVAAGVPQ
JQ01         KLAVIGAGAVGSTLAFAAAQRGIAREIVLEDIAKERVEAEVLDMQHGSSFYPTVSIDGSDDPEICRDADMVVITAGPRQ
JX00        MKVGIVGSGFVGSATAYALVLQGVAREVVLVDLDRKLAQAHAEDILHATPFAHPVWVRSGW YEDLEGARVVIVAAGVAQ
S333         KIALIGAGNVGNSFLYAAMNQGLASEYGIIDINPDFADGNAFDFEDASASLPFPISVSRYEYKDLKDADFIVITAGRPQ

              *  ** *                            ****  *       *       ***  **  *
DEHU        QEGESRLNLVQRNVNIFKFIIPNVVKYSPNCKLLIVSNPVDILTYVAWKISGFPKNRVIGSGCNLDSARFRYLMGERLGV
DECH        QEGESRLNLVQRNVNIFKFIIPNVVKYSPDCKLLIVSNPVDILTYVAWKISGFPKHRVIGSGCNLDSARFRHLMGERLGI
DEDF        QEGESRLNLVQRNVNIFKFIIPNIVKHSPDCIILVVSNPVDVLTYVAWKLSGLPMHRIIGSGCNLDSARFRYLMGERLGV
A382        QEGESRLNLVQRNVNIFKFIIPNIVKYSPNCILLVVSNPVDILTYVAWKLSGLPKHRVIGSGCNLDSARFRYLMSERLGV
A360        IPGETRLNLLQRNVALYRKIVPPVAEHSPDALLLVVSNPVDVLTYVAWKLSGFPASRVIGSGTNLDSSRFRFLIAEHLDV
S224        IQGETRLDLLQRNVALFRKIVPPLAEQSHDALLLVVSNPVDVLTYVAWKLSGFPASRVIGSGTNLDSSRFRFLLAEHLDV
DELB        KPGETRLDLVNKNLKILKSIVDPIVDSGFNGIFLVAANPVDILTYATWKLSGFPKNRVVGSGTSLDTARFRQSIAEMVNV
DEBS        KPGETRLDLVDKNIAIFRSIVSEVMAHGFQGLFLVATNPVDILTYATWKFSGLPHERVIGSGTILDTARFRFLLGEYFSV
B408        KPGESRLDLVNKNLNILSSIVKPVVDSGFDGIFLVAANPVDILTYATWKFSGFPKERVIGSGTSLDSSRLRVALGKQFNV
S368        KPGETRLQLLGRNARVMKEIARNVSKYAPDSIVIVVTNPVDVLTYFFLKESGMDPRKVFGSGTVLDTARLRTLIAQHCGF
JQ01        KPGQSRLELVGATVNILKAIMPNLVKVAPNAIYMLITNPVDIATHVAQKLTGLPENQIFGSGTNLDSARLRFLIAQQTGV
JX00        RPGETRLQLLDRNAQVFADVVPKILKAAPEAVLLIATNPVDVMTQVAYRLSGLPPERVVGSGTILDTARFRALLAQHLLV
S333        KPGETRLELVADNIRIIREIALKVKESGFSGISIIVANPVDIITRAYRDASGFSDQKVIGSGTVLDTARLQFAIAKRAKV

                      * ***                                            **   *  *  *   *
DEHU        HPLSCHGWVLGEHGDSSVPVWSGMNVAGV SLKTLHPDLGTDKDKEQWKEVHKQVVESAYEVIKLKGYTSWAIGLSVADL
DECH        HPLSCHGWIVGQHGDSSVPVWSGVNVAGV SLKALHPDMGTDADKEHWKEVHKQVVDSAYEVIKLKGYTSWAIGLSVADL
DEDF        HSCSCHGWVIGEHGDSVPSVWSGMWNA    LKELHPELGTNKDKQDWKKLHKDVVDSAYEVIKLKGYTSWAIGLSVADL
A382        NSASCHGWIIGEHGDSSVPVWSGVNVAGV GLQSLNPDIGTPKDGEDWKSVHKQVVDSAYEVIKLKGYTSWAIGLSVADL
A360        NAQDVQAYMVGEHGDSSVAIWSSISVGGMPAFKSLR DSHRSFDEAALEGIRRAVVGGAYEVIGLKGYTSWAIGYSVASL
S224        NAQDVQAYMVGEHGDSSVAVWSSVSVAGMPVLKSLQ ESHRCFDEEALEGIRRAVVDSAYEVISLKGYTSWAIGYSVASL
DELB        DARSVHAYIMGEHGDTEFPVWSHANIGGVTIAEWVK  AHPEIKEDKLVKMFEDVRDAAYEIIKLKGATFYGIATALARI
DEBS        APQNVHAYIIGEHGDTELPVWSQAYIGVM PIRKLVESKGEEAQKD LERIFVNVRDAAYQIIEKKGATYYGIAMGLARV
B408        DPRSVDAYIMGEHGDSEFAAYSTATIGTR PVRDV  AKEQGVSDDDLAKLEDGVRNKAYDIINLKGATFYGIGTALMRI
S368        SPRSVHVYVIGEHGDSEVPVWSGAMIGGI PLQNMC QICQKCDSKILENFAEKTKRAAYEIIERKGATHYAIALAVADI
JQ01        NVKNVHAYIAGEHGDSEVPLWESATIGGVPMCDWTPLPGHDPLDADKREEIHQEVKNAAYKIINGKGATNYAIGMSGVDI
JX00        APQSVHAYVVGEHGDSEVLVWSSAQVGGV DLEAFAQARGRALTPDDRLRIDEGVRRAAYRIIEGKGATYYGIGAGLARL
S333        SPNSVQAYVMGEHGDSSFVAYSNIKIAGE CFCAYSKLTGIDSSNYE KELEYPVSRRAYEIINRKRATFYGIGAAIAKI

                                  *                                *      *
DEHU        AESIMKNLRRVHPVSTMIKGLYGIKD  DVFLSVPCILGQ NGISDLVKVTLTSEEEARLKKSADTLWGIQ
DECH        AETIMKNLRRVHPISTAVKGMHGIKD  DVFLSVPCVLGS SGITDVVKMILKPDEEEKIKKSADTLWGIQ
DEDF        AETIMKNLCRVHPVSTMVKDFYGIKD  NVFLSLPCVLND HGISNIVKMKLKPDEEQQLQKSATTLWDIQK
A382        AETILKNLRRVHPVSTHCKGQHGVHD  DVFLSLPCVLGS EGITDIINQTLKKEEEAQVQKSAETLWNVQK
A360        AASLLRDQRRVHPVSVLASGFHGISDGHEVFLSLPARLGR GGILGVAEMDLTEAEAAQLRRSAKTLWENCQ
S224        AASLLRDQRRIHPVSVLARGFHGIPD  GTTSSSACPPRRPRRRPGRREMELTEEEAKRLRRSAKTIWENCQ
DELB        SKAILNDENAVLPLSVYMDGQYGLND   IYIGTPAVINR NGIQNILEIPLTDHEEESMQKSASQLKKVLT
DEBS        TRAILHNENAILTVSAYLDGLYGERD   VYIGVPAVINR NGIREVIEIELNDDEKNRFHHSAATLKSVLA
B408        SKAILRDENAVLPVGAYMDGQYGLND   IYIGTPAIIGG TGLKQIIESPLSADELKKMQDSAATLKKVLN
S368        VESIFFDEKRVLTLSVYLEDYLGVKD   LCISVPVTLGK HGVERILELNLNEEELEAFRKSASILKNAIN
JQ01        IEAVLHDTNRILPVSSMLKDFHGISD   ICMSVPTLLNR QGVNNTINTPVSDKELAALKRSAETLKETAA
JX00        TRAILTDEKGVFTVSLFTPEVEGVEE   VALSLPRILGA RGVEATLYPRLNEEERQALRRSAEILKGAAS
S333        VSNIIKDTKNIMIAGANLRGEYGFHG   VNIGVPVVLGA NGIEKIIEISLNDKEKEKFAKSVAIIDKIYQ
```

FIG. 3. Example of close-pack format that emerges from MULPUB set at 80 columns across. Thirteen L-lactate dehydrogenase sequences were aligned. The four-letter designations are as follows: DEHU, human; DECH, chicken; DEDF, spiny dogfish; A382, sea lamprey; A360, barley; S224, maize; DELB, *Lactobacillus casei;* DEBS, *Bacillus stearothermophilus;* B408, *Lactobacillus plantarum;* S368, *Thermotoga maritima;* JQ01, *Bifidobacterium;* JX00, *Thermus aquaticus;* S333, *Mycoplasma.*

and TREE programs are identical; the only reason to use ALIGN is the time saving if a tree is not the goal. The TREE program goes on to calculate the branching order and branch lengths.

```
TREE seqfile seqfile1 M > seqfile2
```

Moreover, the ARRANGE and TREE programs can be called in a single step by use of the shell file OVERALL, which consists of the following two lines,

```
ARRANGE $1 $4 > $2
TREE $1 $3 $4 > $5
```

The user must specify the starting file with the collected sequences and designate three new files for the storage of outputs:

```
OVERALL seqfile seqfile1 seqfile2 M > seqfile3
```

In this case, the final alignment, percent identities, distance, branching order, and branch lengths appear in seqfile3.

Removing Negative Branch Lengths

There are a number of different ways of reducing a difference matrix composed of m intergroup distances to a phylogenetic tree with n branch lengths. One can solve all the simultaneous equations, for example, or one can use a least squares approach, as suggested by Klotz and Blanken[17] and Li.[18] Although we favor the least squares approach, it does have the flaw that often the best mathematical solution contains one or more negative values, always associated with the inner branch segments of the tree. Most often, the problem can be remedied by simply switching the two taxa (or groups of taxa) that arise at either end of the offending segment (Fig. 4). Many times, however, finding the best arrangement free of negative values is challenging to the point of frustration.

Accordingly, we devised a program that looks for the best matrix-derived tree with no negative segments. It is called NONEG. It works by switching taxa in an iterative manner and testing the result both for the presence of negative segments and the quality of the tree. The latter is measured by a comparison of the initially provided intergroup distances with the final intergroup distances obtained from summing the appropriate branchlengths and is given as a percent standard deviation. The input data

[17] L. C. Klotz and R. L. Blanken, *J. Theor. Biol.* **91,** 261 (1981).
[18] W.-H. Li, *Proc. Natl. Acad. Sci. U.S.A.* **78,** 1085 (1981).

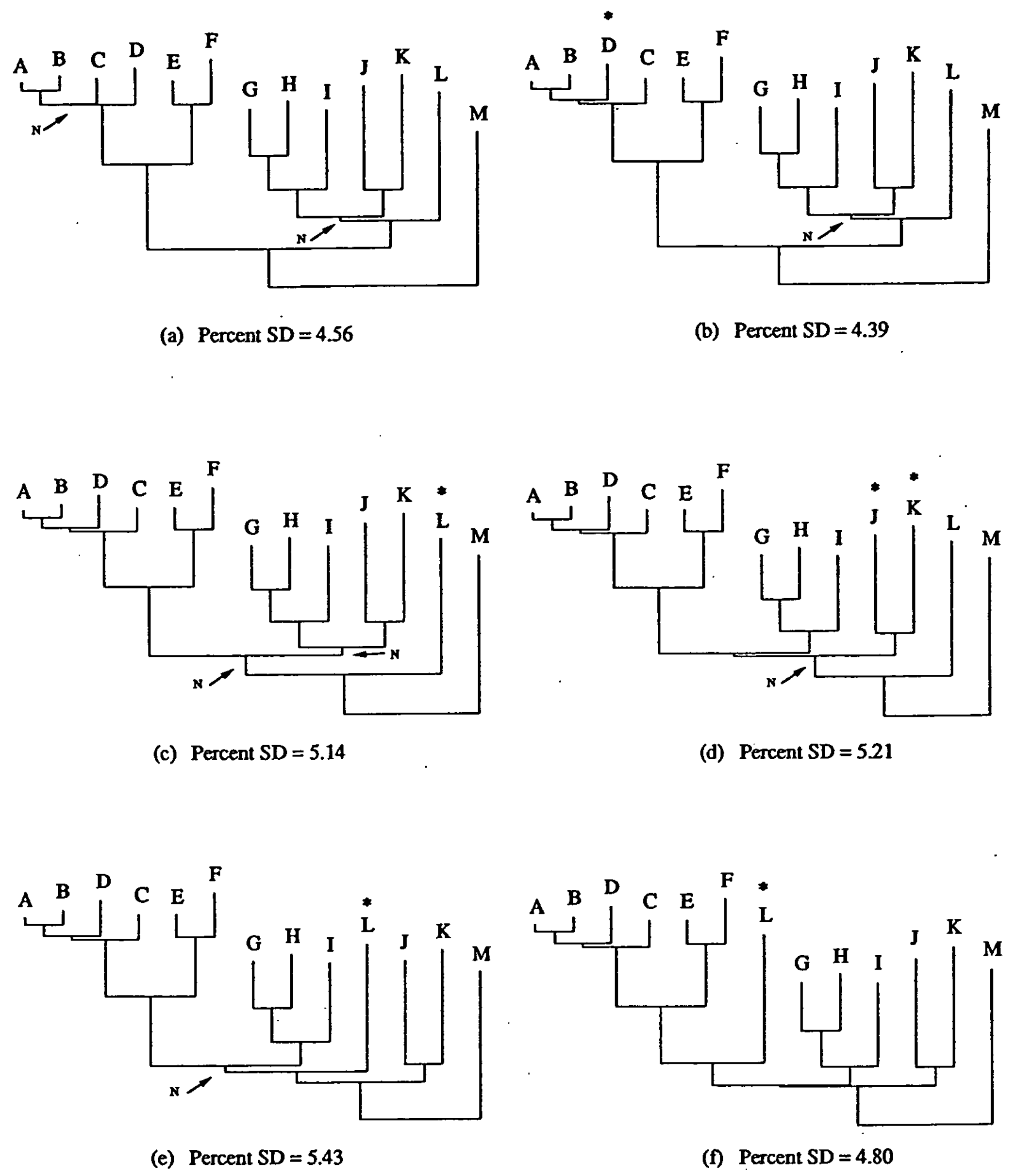


FIG. 4. Illustration of branch swapping that occurs during the elimination of negative branch lengths. In the original tree (a) the arrows indicate two negative branch lengths (for drawing purposes, they are shown as having positive lengths). Asterisks (*) denote taxa involved in switching at various steps. For example, switching the taxa C and D eliminates the first of the negative branch lengths (b) and at the same time improves the overall tree as evidenced by the percent standard deviation dropping from 4.56 to 4.39. Subsequent switchings (c, d, or e) actually make the tree worse without eliminating the final negative value, until finally (f) a solution is reached.

for NONEG appear automatically in a file called ALBR (for alternate branching) every time the program TREE is run. The protocol is simply

```
NONEG albr > newfile
```

The newfile contains the initial, intermediate, and final branching orders, along with their respective percent standard deviations.

Bootstrap Analysis

The purpose of making a phylogenetic tree is to provide an estimate of the underlying biological relationships. Because the tree is derived from a multiple sequence alignment, which in turn depends on fluctuations in the real biological world, and to a lesser extent on systematic variables such as the gap penalty and the scoring matrix, the information is expected to have inherent variability. Felsenstein applied the bootstrapping method to phylogenetic trees in an effort to estimate these statistical fluctuations.[19] The method has become a popular way of expressing confidence about the branching order on a node by node basis.

In our version, bootstrapping is invoked by substituting the program BTREE for TREE. The bootstrap itself is performed as follows: the alignment and branching order are found as described above. Then, columns of residues are sampled randomly from the multiple alignment, with replacement. Columns in which gaps (represented as neutral element X) occur in more than half of the positions, and erratic overhangs, if present, are ignored. This sample selection process continues until the lengths of the sequences are the same as the originally aligned sequences. Equations (1), (2), and (3) are used to determine the distance matrix. The best branching order with all positive branch lengths is then saved. The procedure is repeated at least 100 times. Finally, all the branching orders are compared with the original branching order on a node by node basis. The number of times a cluster (around a node) is the same as in the original tree is tabulated; the results are expressed in percentages and placed at the nodes.

Tree Draw Interface

The program TREE produces two hidden files, ALBR and TOMAC. Both contain a branching order and a set of branch lengths, but their formats are different. As noted above, ALBR (alternate branch lengths) is used as an input file for NONEG whenever one or more negative branch

[19] J. Felsenstein, *Evolution* **39,** 783 (1985).

```
((((AB)C)D)(EF))((((GH)I)(JK))L))M
     % s.d. (obs. vs calc.) =    4.56

((((AB)D)C)(EF))((((GH)I)(JK))L))M
     % s.d. (obs. vs calc.) =    4.39

(((((AB)D)C)(EF))(((GH)I)(JK)))L)M
     % s.d. (obs. vs calc.) =    5.14

((((((AB)D)C)(EF))((GH)I))(JK))L)M
     % s.d. (obs. vs calc.) =    5.21

((((((AB)D)C)(EF))((GH)I))L)(JK))M
     % s.d. (obs. vs calc.) =    5.43

((((((AB)D)C)(EF))L)((GH)I))(JK))M

Branch lengths are (r1,r2,r3,r4,r5):
                   (r6,r7,........):

    1.96      3.74      3.05      9.65      0.99
    6.91     15.76      6.24     11.94     16.38
   14.84     33.56      5.91     12.79     14.85
    9.28     21.67      4.94      0.44     28.33
   31.22      5.01     50.53

     % s.d. (obs. vs calc.) =    4.80
```

FIG. 5. Output of NONEG using the L-lactate dehydrogenase tree example.

lengths appear. It depicts the branching order with a set of consecutive lines that describe clusters:

```
ABCDEFG
DE
*
```

The asterisk denotes the end of branching order information and signifies the beginning of the array of branch lengths.

TOMAC, on the other hand, is the input file to use on the way to tree drawing; it is automatically updated by NONEG. The file begins with the convenient one-line notation of the branching order as described by Fitch[20] and is followed directly by the branch lengths, data that can be read by the program SETREE. SETREE is an interactive program in which the user has a choice of using the default four-letter code or of specifying a new name for each taxon (limited to 20 characters; no spaces or parentheses allowed). The output file, whatever it is named, will contain the phylogenetic tree information in a format that can be read by a hypercard tree drawing program called TREEDRAW DECK that has been adapted from the

[20] W. M. Fitch, *Am. Nat.* **111,** 223 (1977).

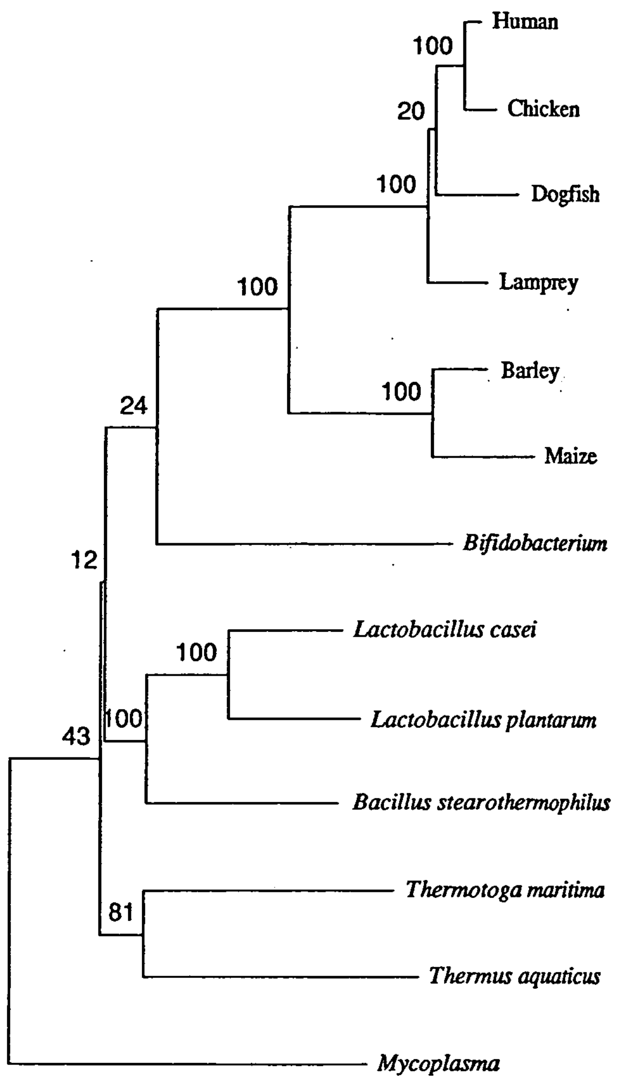


FIG. 6. Bootstrap analysis (100 trials) of 13 L-lactate dehydrogenase sequences. The analysis was restricted to trees with all positive branch lengths.

Felsenstein PHYLIP programs[21] by D. G. Gilbert of Indiana University. This program can be downloaded from the Internet (dgilbert@iubio.bio.indiana.edu) and installed on a suitable microcomputer (our version happens to interface with a MacIntosh). All of the above steps are summarized in Fig. 1.

Example

An illustration of these programs can be afforded by a consideration of a set of L-lactate dehydrogenase (LDH) sequences. From the PIR data-

[21] J. Felsenstein's PHYLIP programs can be obtained by anonymous ftp from 128.95.12.41.

base, 13 LDH sequences were taken, run through the program FORMAT, and then catenated into a file called TEST1, without regard to order. The program ARRANGE was used to calculate first all the binary alignments and put the sequences in an approximate order, and then to proceed directly to making the multiple alignment and finding a branching order.

```
OVERALL TEST1 TEST2 TEST3 D > TEST4
```

The output file TEST4 contains the aligned sequences, tables of similarity and distance, a branching order, and a set of branch lengths determined by the method of least squares. As it happens, in this instance, the table of branch lengths contains two negative values (Fig. 4a). Accordingly, the program NONEG was used to find a valid branching order.

```
NONEG ALBR > TEST5
```

The output (in TEST5) shows the beginning, intermediate, and final branching orders and their respective percent standard deviations (Fig. 5).

The SETREE program is then used to generate a file for tree draw.

```
SETREE tomac > tomac1
```

Finally, the bootstrap version of the program was conducted separately:

```
BTREE seqfile seqfile1 D > resultfile
CLUS tomac1 rboot > rboot1
```

It must be underscored that the bootstrapping is restricted to solutions with no negative branch lengths, a necessary condition if the likelihoods are to have any meaning. This condition is often ignored by other bootstrapping procedures involving amino acid alignments made by matrix methods. The results are stored in a hidden file named rboot. A full calculation generally consists of 100 bootstrap trials. When rboot is analyzed by the program CLUS, the number of times (for 100, it is the percentage) a node is in agreement with that in the starting tree is shown on the phylogenetic tree (Fig. 6).

Program Availability

All the programs described in this chapter are available by anonymous ftp from juno.ucsd.edu. After logging in with any identifying name as a password, claimants should go to the directory progs/.

[22] Using CLUSTAL for Multiple Sequence Alignments

By Desmond G. Higgins, Julie D. Thompson, and Toby J. Gibson

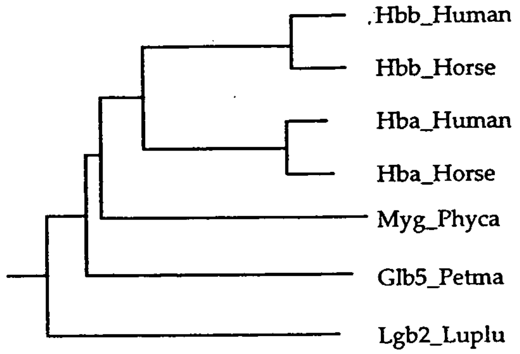
Introduction

The simultaneous alignment of many nucleotide or amino acid sequences is now one of the commonest tasks in computational molecular biology. It forms a common prelude to phylogenetic analysis of sequences, the prediction of secondary structure (DNA or protein), the detection of homology between newly sequenced genes and existing sequence families, the demonstration of homology in multigene families, and the finding of candidate primers for PCR (polymerase chain reaction). For such a widely required analysis, it is surprising how long it took for practical methods to appear. Up until 1987, it was standard practice to construct multiple alignments manually. This is very tedious and error prone. The basic problem was that direct extensions of the standard dynamic programming approach for the alignment of two sequences were computationally impossible for more than three real sequences. Some methods had been invented[1-4] based on trying to find alignment blocks or on iterating toward a consensus sequence, for example, but these were not very widely used. The first practical methods for the sensitive multiple alignment of many protein sequences appeared in 1987 and 1988[5-10] and were based on an original idea by David Sankoff.[11] The idea is to exploit the fact that groups of sequences are phylogenetically related (if they can be aligned, there is usually an underlying phylogenetic tree). This approach is commonly referred to as progressive alignment.[6] Most of the automatic multiple alignments that appear in the current literature are carried out using this approach.

[1] W. Bains, *Nucleic Acids Res.* **14,** 159 (1986).
[2] E. Sobel and H. M. Martinez, *Nucleic Acids Res.* **14,** 363 (1986).
[3] D. J. Bacon and W. F. Anderson, *J. Mol. Biol.* **191,** 153 (1986).
[4] M. S. Johnson and R. F. Doolittle, *J. Mol. Evol.* **23,** 267 (1986).
[5] W. R. Taylor, *CABIOS* **3,** 81 (1987).
[6] D.-F. Feng and R. F. Doolittle, *J. Mol. Evol.* **25,** 351 (1987).
[7] G. J. Barton and M. J. E. Sternberg, *J. Mol. Biol.* **198,** 327 (1987).
[8] W. R. Taylor, *J. Mol. Evol.* **28,** 161 (1988).
[9] D. G. Higgins and P. M. Sharp, *Gene* **73,** 237 (1988).
[10] F. Corpet, *Nucleic Acids Res.* **16,** 10881 (1988).
[11] D. Sankoff, *SIAM J. Appl. Math.* **78,** 35 (1975).

Hbb_Human	1	-					
Hbb_Horse	2	.17	-				
Hba_Human	3	.59	.60	-			
Hba_Horse	4	.59	.59	.13	-		
Myg_Phyca	5	.77	.77	.75	.75	-	
Glb5_Petma	6	.81	.82	.73	.74	.80	-
Lgb2_Luplu	7	.87	.86	.86	.88	.93	.90
		1	2	3	4	5	6

Pairwise alignment: Calculate distance matrix



Rooted Neighbor Joining tree (guide tree)

```
--------VHLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDLST
--------VQLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDLSN
---------VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLS--
---------VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLS--
---------VLSEGEWQLVLHVWAKVEADVAGHGQDILIRLFKSHPETLEKFDRFKHLKT
PIVDTGSVAPLSAAEKTKIRSAWAPVYSTYETSGVDILVKFFTSTPAAQEFFPKFKGLTT
--------GALTESQAALVKSSWEEFNANIPKHTHRFFILVLEIAPAAKDLFSFLKGTSE
          *.      .   *                     * .   *

PDAVMGNPKVKAHGKKVLGAFSDGLAHLD-----NLKGTFATLSELHCDKLHVDPENFRL
PGAVMGNPKVKAHGKKVLHSFGEGVHHLD-----NLKGTFAALSELHCDKLHVDPENFRL
----HGSAQVKGHGKKVADALTNAVAHVD-----DMPNALSALSDLHAHKLRVDPVNFKL
----HGSAQVKAHGKKVGDALTLAVGHLD-----DLPGALSNLSDLHAHKLRVDPVNFKL
EAEMKASEDLKKHGVTVLTALGAILKKKG-----HHEAELKPLAQSHATKHKIPIKYLEF
ADQLKKSADVRWHAERIINAVNDAVASMDDT--EKMSMKLRDLSGKHAKSFQVDPQYFKV
VP--QNNPELQAHAGKVFKLVYEAAIQLQVTGVVVTDATLKNLGSVHVSKG-VADAHFPV
         .. *   .                         *   *     .

LGNVLVCVLAHHFGKEFTPPVQAAYQKVVAGVANALAHKYH------
LGNVLVVVLARHFGKDFTPELQASYQKVVAGVANALAHKYH------
LSHCLLVTLAAHLPAEFTPAVHASLDKFLASVSTVLTSKYR------
LSHCLLSTLAVHLPNDFTPAVHASLDKFLSSVSTVLTSKYR------
ISEAIIHVLHSRHPGDFGADAQGAMNKALELFRKDIAAKYKELGYQG
LAAVIADTVAAG---------DAGFEKLMSMICILLRSAY-------
VKEAILKTIKEVVGAKWSEELNSAWTIAYDELAIVIKKEMNDAA---
.   .   .                           .
```

Progressive alignment: Align following the guide tree

Progressive alignment involves making initial guesses as to the phylogenetic relatedness of the sequences and using the branching order in an initial phylogenetic tree to align larger and larger groups of sequences. Start by aligning the most closely related pairs of sequences using dynamic programming and gradually align these groups together, keeping gaps that appear in early alignments fixed. At each stage, align two sequences or one sequence to an existing subalignment or align two subalignments. There are now many variations on the approach. The initial tree may be modified as the procedure progresses[12]; alternative branching orders may be tested at each stage[6]; the initial tree may be replaced by a tree calculated from the fully aligned sequences and the procedure iterated until the alignment (or tree) converges[10]; sequences may be aligned together in a nonphylogenetic manner but rather by adding sequences one at a time to a growing alignment.[7] Further, one can use different strategies for aligning the groups of prealigned sequences. For example, one can base the alignment only on the alignment of the two most closely related sequences (one from each alignment).[5] In most current implementations, the subalignments are aligned together using information from all of the constituent sequences with an extension of the profile alignment approach.[13] An example of the approach is shown in Fig. 1 for the alignment of seven globin sequences of known tertiary structure.

Algorithmically, the progressive approach is considered to be "heuristic" insofar as the method is not guaranteed to produce alignments with any particular mathematical property such as maximum alignment score. The method is, however, soundly based biologically and has the great

[12] J. Hein, *Mol. Biol. Evol.* **6,** 649 (1989).
[13] M. Gribskov, A. D. McLachlan, and D. Eisenberg, *Proc. Natl. Acad. Sci. U.S.A.* **84,** 4355 (1987).

FIG. 1. Outline of the progressive multiple alignment approach in CLUSTAL W. Seven globin sequences of known tertiary structure are used (the sequence names are identifiers from the SWISS-PROT sequence database). The alignment was carried out using default parameters except that the Dayhoff PAM weight matrices [M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 345. National Biomedical Research Foundation, Washington, D.C., 1978] were used instead of BLOSUM [S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 10915 (1992)]. The approximate positions of the seven α helices, common to all sequences, are shown as boxes. The alignment is carried out by aligning the two α-globins (Hba_Human and Hba_Horse) together; then the two β-globins are aligned (Hbb_Human and Hbb_Horse); then the two aligned α-globins are aligned to the two aligned β-globins; finally, the whale myoglobin (Myg_Phyca), the lamprey cyanohemoglobin (Glb5_Petma), and the lupine leghemoglobin (Lgb2_Luplu) are aligned one at a time to the growing alignment.

advantage of speed and simplicity. One can align hundreds of sequences, even on personal computers. More importantly, the sensitivity of the approach, as judged by the ability to align distantly related sequences, is very high. In simple cases, it is common to derive alignments which are impossible to improve by eye. In these cases, the method can be said to be a satisfactory replacement for manual alignment. In more difficult cases, the method usually gives a useful starting point for further refinement. In this chapter, we describe some of the problems of progressive multiple alignment and some simple modifications which, we believe, greatly improve the sensitivity for difficult protein alignments. All of the methods described here are freely available in a computer program called CLUSTAL W which can be run under a wide variety of operating systems.

Problems with Progressive Alignment

There are two obvious and interrelated problems inherent in the progressive alignment approach: (1) the local minimum problem and (2) the parameter choice problem. The local minimum problem stems from the "greedy" nature of the algorithm. Every time an alignment is carried out, some proportion of the residues will be misaligned. This proportion will be very small (or nonexistent) for very closely related proteins (e.g., at least 50% identical) but will increase as more and more divergent sequences are used. Any mistakes that appear during early alignments in a progressive multiple alignment cannot be corrected later as new sequence information is added. If the data set contains sequences of different degrees of divergence, the first alignments may be very accurate, and by the time the most diverged sequences are aligned, some information about gap frequency and residue conservation at each position will be available. In this case, the progressive approach may work very well. In other cases, however, if the first alignments are not correct they cannot be corrected later.

It is commonly argued that the local minimum problems stems largely from errors in the branching order of the initial phylogenetic tree. Consequently, many authors have devoted great effort to investigating alternative topologies or advocating particular methods of phylogenetic analysis. For example, previous versions of CLUSTAL[9,14,15] have been criticized[16] for using UPGMA[17] to generate initial trees as UPGMA is notorious for giving

[14] D. G. Higgins and P. M. Sharp, *CABIOS* **5,** 151 (1989).
[15] D. G. Higgins, A. J. Bleasby, and R. Fuchs, *CABIOS* **8,** 189 (1992).
[16] C.-B. Stewart, *Nature* (*London*) **367,** 26 (1994).
[17] P. H. A. Sneath and R. R. Sokal, "Numerical Taxonomy." Freeman, San Francisco, 1973.

incorrect branching orders when rates of substitution vary greatly in different lineages. This criticism is erroneous. If the alignment is simple enough, almost any tree will give the correct alignment. If the alignment is sufficiently difficult, almost any tree will give the wrong alignment. There is no one-to-one correspondence between having the correct tree topology and getting the right alignment. The better the tree, then the better the chances of getting a good alignment, but there are no guarantees. Even if UPGMA is not ideal for obtaining correct tree topologies, it can be argued that it is still very useful for alignment purposes. At each stage in the alignment process, align the most similar remaining sequences or subalignments so as to minimize the alignment errors at that step. UPGMA gives this property automatically. Nonetheless, we now provide the neighbor-joining method[18] for making initial trees because it seems to provide more reliable tree topologies and gives better estimates of tree branch lengths which we use to weight sequences and adjust the alignment parameters dynamically.

The local minimum problem is intrinsic to progressive alignment, and we do not provide any direct solutions. The only way to correct it is to use an overall measure of multiple alignment quality and find the alignment which maximizes this measure. This can be done directly for small numbers of sequences using the program MSA[19] but is, for now, uncomputable for more than about seven sequences. MSA computes approximate bounds for the location of the best pathway through the *N*-dimensional (for *N* sequences) dynamic programming array and then carries out full dynamic programming in this restricted area. It may be possible to use stochastic or iterative optimization procedures[20,21] to optimize multiple alignments of many sequences in the future. Even if practical solutions are found, we believe that the parameter choice problem, described below, is just as important and will still preclude high accuracy alignments if it is not addressed.

The parameter choice problem stems from using just one set of parameters (normally an amino acid substitution matrix and two gap penalties) and hoping that these will be appropriate over all parts of all the sequences to be aligned. If the sequences are very similar, almost any reasonable parameters will give a good alignment. For highly divergent sequences, however, the exact choice of parameters may have a great effect on align-

[18] N. Saitou and M. Nei, *Mol. Biol. Evol.* **4,** 406 (1987).
[19] D. Lipman, S. F. Altschul, and Kececioglu, *J. Proc. Natl. Acad. Sci. U.S.A.* **86,** 4412 (1989).
[20] O. Gotoh, *CABIOS* **9,** 361 (1993).
[21] C. E. Lawrence, S. F. Altschul, M. S. Boguski, J. S. Liu, A. F. Neuwald, and J. C. Wooton, *Science* **262,** 208 (1993).

ment quality. Different weight matrices from the well-known PAM[22] or BLOSUM[23] series are appropriate for aligning sequences of different evolutionary distances. For very similar sequences, even an identity matrix will provide sensible alignments, but for sequences in the so-called twilight zone[24] (sequences of roughly 20–25% identity), the exact values given to each type of substitution may be critical. Further, there is no particular reason why the same gap penalties should work equally well at all positions in an alignment. Gaps tend to occur far more often between the main secondary structure elements of α helices and β strands than within.[25] With the latest CLUSTAL program (CLUSTAL W), we attempt to attack this problem by computing position-specific gap opening and extension penalties as the alignment proceeds. We also use different amino acid weight matrices; "hard" ones for closely related sequences and "softer" ones for more divergent sequences.

We believe that the correct use of parameters at different positions is important. We provide simple heuristic methods for doing this which seem to work well in difficult test cases. Other authors have attacked the problem using more systematic methods such as hidden Markov models[26] or have exploited structural information when the structure of one or more of the sequences is known.[27]

CLUSTAL W

CLUSTAL W[28] is derived directly from the CLUSTAL[9,14] and CLUSTAL V series of programs. The "W" in the name stands for "weighting" as we now give different weights to sequences and parameters at different positions in alignments. The main feature of the old programs was the ability to align many sequences quickly, even on a personal computer, with a minimal sacrifice in sensitivity. The new program offers similar speed as before (CLUSTAL W is in fact slower than the older programs, but, fortunately, advances in the power of personal computers and workstations have canceled this out) but provides a number of new features and appears

[22] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 345. National Biomedical Research Foundation, Washington, D.C., 1978.

[23] S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 10915 (1992).

[24] R. F. Doolittle, "URFs and ORFs: A Primer on How to Analyze Derived Amino Acid Sequences." University Science Books, Mill Valley, California, 1987.

[25] S. Pascarella and P. Argos, *J. Mol. Biol.* **224,** 461 (1992).

[26] A. Krogh, M. Brown, S. Mian, K. Sjölander, and D. Haussler, *J. Mol. Biol.* **235,** 1501 (1994).

[27] G. J. Barton and M. J. E. Sternberg, *Protein Eng.* **1,** 89 (1987).

[28] J. D. Thompson, D. G. Higgins, and T. J. Gibson, *Nucleic Acids Res.* **22,** 4673 (1994).

to be more sensitive for difficult protein alignments. The main new features are (1) support for more file formats for trees, sequence data sets, and alignments; (2) optional, full dynamic programming alignments for estimating the initial pairwise distances between all the sequences; (3) neighbor-joining[18] trees for the initial guide trees, used to guide the progressive alignments; (4) sequence weighting to correct for unequal sampling of sequences at different evolutionary distances; (5) dynamic calculation of sequence- and position-specific gap penalties as the alignment procedes; (6) the use of different weight matrices for different alignments; and (7) improved facilities for adding new sequences to an existing alignment.

The source code of CLUSTAL W, version 1.5 (April 1995), is available free of charge from the EMBL E-mail file server or by anonymous ftp to the EMBL ftp server. Compiled versions are also available for MS-DOS and Macintosh computers. With the Macintosh version, we also supply the NJplot program of Manolo Gouy (University of Lyon) which allows for the graphical display and manipulation of phylogenetic trees.

Use ftp to connect to ftp.ebi.ac.uk and give user name anonymous and full E-mail address as password. The four versions are as follows:

/pub/software/vax/clusalw.uue	uuencoded ZIP archive for VMS
/pub/software/unix/clustalw.tar.Z	compressed tar archive for UNIX
/pub/software/dos/clustal$.exe	self-extracting archive for MS-DOS
/pub/software/mac/clustalw.sea.hqx	Binhex encoded self-extracting archive

The MS-DOS and UNIX versions should be transferred in binary mode; the VMS and Mac versions in ASCII.

Position-Specific Gap Penalties

Here we give a brief summary of the methods used to calculate position-specific gap penalties for protein alignments. Two gap penalties are used initially: a gap opening penalty (GOP) and a gap extension penalty (GEP). Traditionally, one will choose values for these parameters before alignment and use the same values for all sequences. The exact values used are usually the default values offered by the software, which are often chosen empirically by the software authors by trial and error for a given amino acid weight matrix. In a simple world where one knew the positions of all secondary structure elements (α helices and β strands) in all or some of the sequences, one could increase the GOP (and GEP) at each position inside a helix or strand and decrease it between them.[27] This would force gaps to occur most often in loop regions, which is what is observed in practice with test cases from protein structure superposition.[25] One could be more sophisticated and make the GOP highest at the center of helices and strands and reduce it at the edge, allowing some gaps to occur at the ends of secondary structure elements. If one does not know the secondary structure of the sequences, as is normally the case, this is not possible. In CLUSTAL W, we use a set of very simple rules to help modify the GOP and GEP at each position in a sequence or prealigned group of sequences, depending on the residues that occur at each position and the frequency of gaps at each position. These rules are simple heuristics that seem to work very well in practice, although it should be possible to derive similar rules with greater statistical or mathematical validity.

Before any two sequences or prealigned groups of sequences are aligned, we calculate initial values for the GOP and GEP as functions of the amino acid weight matrix to be used, the sequence (or alignment) lengths, and the divergence between the sequences. The values for GOP and GEP are set from a user-controlled menu (defaults are offered) and then modified as follows:

$$\mathrm{GOP} \rightarrow A * B * \{\mathrm{GOP} + \log[\min(N, M)]\} \tag{1}$$

where N and M are the lengths of the sequences to be aligned, A is the average value for a mismatch in the amino acid weight matrix, and B is the percent identity of the two sequences. The GEP is then modified using the following formula:

$$\mathrm{GEP} \rightarrow \mathrm{GEP} * [1.0 + |\log(N/M)|] \tag{2}$$

where N and M are, again, the lengths of the two sequences.

The overall effect of these transformations is to allow for the use of different weight matrices with sequences of different degrees of divergence and to try to correct for some side effects of using sequences of different lengths. If the sequences are greatly different in length (as measured by the ratio N/M), the GEP is increased to try to inhibit the appearance of too many long gaps in the shorter sequence.

Next, tables of GOP and GEP values are calculated for each of the two sequences or groups of sequences to be aligned, one GOP and GEP for each position. Initially, these values are all the same, as calculated using Eqs. (1) and (2) above. These are then modified at each position using four rules. The overall aim is to encourage gaps to occur in likely loop regions. Informally, the rules are (1) use lower gap penalties at positions where gaps already occur; (2) increase gap penalties adjacent to positions where gaps already occur; (3) reduce gap penalties where stretches of hydrophilic residues occur; and (4) increase or decrease gap penalties using tables of the observed frequencies of gaps adjacent to each of the 20 amino acids.[25]

If there are gaps at a position in a group of prealigned sequences (this rule and the following one do not apply to single sequences), then the GOP is reduced in proportion to the number of sequences with a gap at that position and the GEP is lowered by one-half. The new GOP is calculated as

$$\text{GOP} \rightarrow \text{GOP} * 0.3 * (W/N) \tag{3}$$

where W is the number of sequences without a gap at the position and N is the number of sequences.

If a position contains no gaps but is within eight residues of an existing gap (this value of 8 can be changed from a menu), the GOP is increased as follows:

$$\text{GOP} \rightarrow \text{GOP} * \{2 + [8 - (D) *2]/8\} \tag{4}$$

where D is the distance from the gap.

A run of five (this number can be changed from a menu) consecutive, hydrophilic residues is considered to be a hydrophilic stretch. The residues that are considered to be hydrophilic are conservatively set to D, E, G, K, N, Q, P, R, and S by default but can be changed by the user. Any positions with no gaps that are spanned by such a stretch of residues get the GOP reduced by one-third.

In Table I, we list 20 residue-specific gap propensity values. These are derived from the observed frequencies of gaps adjacent to each residue in

TABLE I
RESIDUE-SPECIFIC GAP OPENING PENALTY FACTORS[a]

Residue	Penalty	Residue	Penalty
A	1.13	M	1.29
C	1.13	N	0.63
D	0.96	P	0.74
E	1.31	Q	1.07
F	1.20	R	0.72
G	0.61	S	0.76
H	1.00	T	0.89
I	1.32	V	1.25
K	0.96	Y	1.00
L	1.21	W	1.23

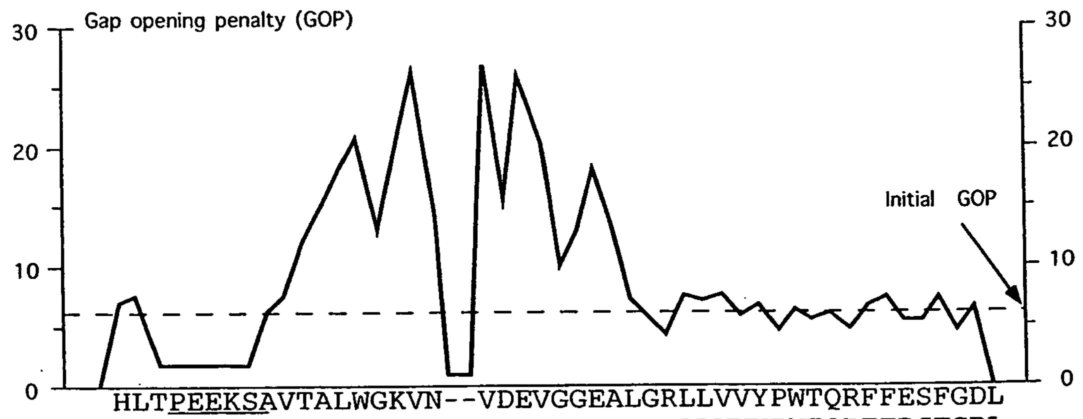
[a] These values are derived from the observed frequencies of gaps adjacent to each residue in alignments of sequences of known tertiary structure.[25] The values were transformed from the published values such that the bigger the number, the less likely a gap is to occur adjacent to that residue. The numbers are then used as simple multiplication factors to modify gap opening penalties, normalized around a value of 1.0 for histidine.

alignments of sequences with known three-dimensional structure.[25] If a position does not contain a gap or a hydrophilic stretch, then the values in Table I are used as simple multiplication factors to increase or decrease the GOP; for example, a position with only glycine will get a reduced GOP (multiplied by 0.61), whereas a position with only methionine will get an increased GOP (multiplied by 1.29). If there is a mixture of residues at a position, then the multiplication factor is the average of those for each residue, one from each sequence.

The overall effect of the four rules can be seen in Fig. 2 on a small stretch of alignment from four globin sequences. The GOP is highest adjacent to a gap and lowest at a gap position or where hydrophilic runs occur. These rules are most useful when there are already some sequences correctly aligned. Then, new gaps will tend to concentrate in areas where gaps already occur and will promote a blocklike appearance in the final alignment. For the first alignments, before any sequences are aligned, there are no gaps yet and the first two rules above cannot be used; only the residue-specific gap frequency rule and hydrophilic stretch rules can be used.

Sequence Weighting

In most real data sets of protein sequences, it is common to have unequal sampling of sequences at different evolutionary distances. Frequently, there



```
HLTPEEKSAVTALWGKVN--VDEVGGEALGRLLVVYPWTQRFFESFGDL
QLSGEEKAAVLALWDKVN--EEEVGGEALGRLLVVYPWTQRFFDSFGDL
VLSPADKTNVKAAWGKVGAHAGEYGAEALERMFLSFPTTKTYFPHFDLS
VLSAADKTNVKAAWSKVGGHAGEYGAEALERMFLGFPTTKTYFPHFDLS
```

FIG. 2. Illustration of the effect of modifying the gap opening penalty (GOP) on a stretch of globin alignment from Fig. 1. The initial GOP is shown as a dotted line, and the position-specific GOP values are plotted along the alignment. The GOP is lowest at positions with gaps and where hydrophilic stretches occur (two such stretches are underlined). The GOP is highest within eight residues of gaps. The rest of the variation is caused by the residue-specific factors from Table I.

are clusters of closely related sequences and small numbers of highly diverged ones. For example, in the data set illustrated in Fig. 1, there are two mammalian α-globins and two β-globins. These pairs are almost identical and provide little extra information for alignment purposes. Traditionally, during multiple alignment or when using aligned sequences as profiles for database searching, one gives equal weight to all sequences. In the globin example, this means that the two α-globins are given as much weight each as the more diverged leghemoglobin. Several authors have addressed the problem of sequence weighting in order to correct for this effect[29]; closely related sequences are down-weighted, while relatively more distant ones receive greater weight. In profile searches, this has been shown to increase the sensitivity of the search as measured by the ability to detect distant relatives of the sequences in the profile.[30-32] Sequence weights were first used for multiple alignments by Vingron and Argos[33] and are also used in the MSA[19] program.

There are several methods available for sequence weighting. For our purposes, it must be possible to derive weights for unaligned sequences as the weights will be used to help arrive at the multiple alignment itself. This prevents the use of methods that use multiple alignments as input. We use a simple tree-based method that only requires a tree describing the rough relatedness of the sequences.[30] The guide trees, used to guide the multiple alignment, provide this. The method requires a rooted tree with branch lengths. As a first approximation, the weights are calculated as the distance of each sequence from the root of the tree. This gives increased weight to the most diverged sequences and less to the more conserved ones. Second, if two sequences (or groups of sequences) share a common internal branch, the length of the internal branch is shared when deriving the weights. This automatically downweights related sequences in proportion to the degree of relatedness. An example is shown in Fig. 3, using the guide tree from Fig. 1. Here the leghemoglobin (Lgb2_Luplu) gets a weight of 0.442, which is equal to the length of the branch from the root to it. The human α-globin (Hba_Human) receives a weight equal to the length of branch leading to it that is not shared by any other sequences (0.055) plus one-half the length of the branch shared with the horse α-globin (0.219/2) plus one-quarter the length of the branch shared by all four hemoglobins (0.061/4) plus one-fifth the branch shared between the hemoglobins and the myo-

[29] M. Vingron and P. R. Sibbald, *Proc. Natl. Acad. Sci. U.S.A.* **90,** 8777 (1993).
[30] J. D. Thompson, D. G. Higgins, and T. J. Gibson, *CABIOS* **10,** 19 (1994).
[31] R. Lüthy, I. Xenarios, and P. Bucher, *Protein Sci.* **3,** 139 (1994).
[32] S. Henikoff and J. G. Henikoff, *J. Mol. Biol.* **243,** 574 (1994).
[33] M. Vingron and P. Argos, *CABIOS* **5,** 115 (1989).

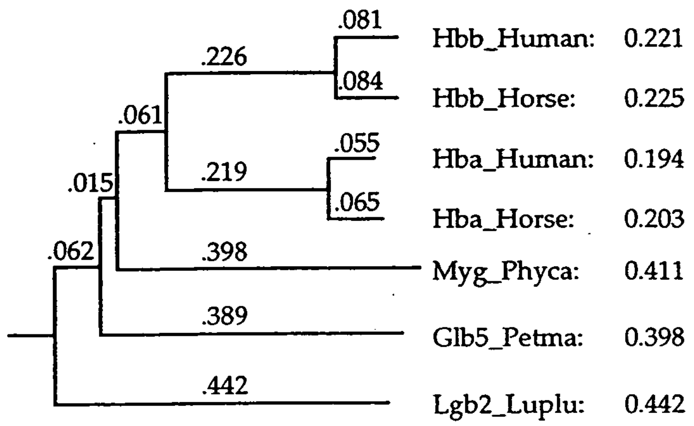


FIG. 3. Sequence weights for the seven globin sequences from Fig. 1. A rooted neighbor-joining tree is shown with branch lengths. The weights are shown for each sequence before normalization (the weights are normalized so as to make the largest equal to 1.0).

globin (0.015/5) plus one-sixth the branch shared by all the vertebrate globins (0.062/6). This gives a total weight of 0.194 before normalization. All weights are normalized to make the largest weight equal to one.

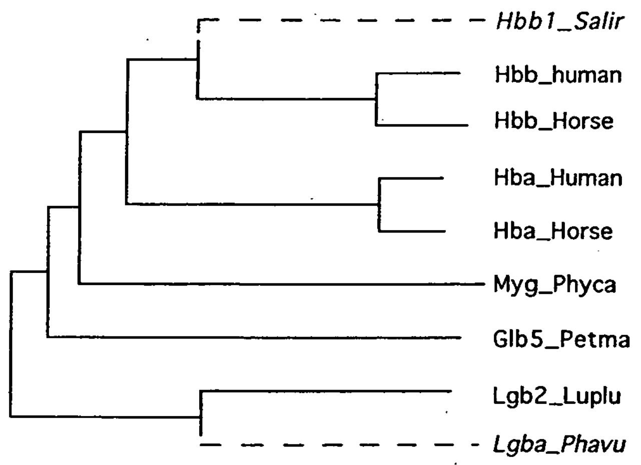
These weights are intuitive, simple to calculate, and have been shown to be useful in increasing the sensitivity of profile searches.[30,32] With multiple alignment, they are used to give different weight to the contributions of different sequences to the alignment scores when aligning two subalignments or when aligning a sequence to a subalignment. The weights are used as simple multiplication factors when calculating the alignment score between two positions.

Weights for Adding New Sequences to Existing Alignment

Sequence weights are also useful when adding new sequences to an existing alignment. In CLUSTAL W, we provide facilities to do this in three ways: (1) add a single sequence to an alignment; (2) add a set of new sequences one at a time to an alignment; (3) align two existing alignments. With methods 1 and 2, there is a further use of sequence weighting. If the sequence to be aligned is much closer to some of the sequences in the alignment (the new sequence can be said to go "inside" the underlying tree), then one can exploit this to give extra weight to the most closely related examples (most closely related to the new sequence). This will help to make sure that the placement of new gaps is most influenced by the close relatives rather than distantly related sequences. The weights that achieve this effect are, again, very simple to derive. Pairwise alignments and resulting simple alignment distances (mean number of differences per site, ignoring gaps and not corrected for multiple hits) are calculated between the new sequence and each sequence in the old alignment. Then, new weights are calculated for each sequence in the alignment as 1.0 minus the distance from the new sequence. This has the effect of giving a weight of 1.0 to identical sequences (identical to the new sequence) and a low weight to distant relatives. Finally, these weights are multiplied with the original tree weights for each sequence in order to combine the properties of the two types of weights, and the weights are normalized so as to sum to 1.

An example is shown in Fig. 4 where two globins are added to the previous globin alignment. The two sequences are compared to each of the sequences in the alignment. The sequences are added to the alignment such that the most similar ones (most similar, on average, to the sequences in the alignment) are added first. In this case, the trout β-globin (Hbb1_Salir) is added first using the weights shown in Fig. 4. These weights give increased weight to the most closely related sequences (closest to the new sequence) but also downweight sequences that are closely related to each other. Then, a new set of weights are calculated for the addition of the new leghemoglobin sequence (Lgba_Phavu), including a weight for the trout β-globin.

We have not examined the properties of these new weights for adding sequences to an alignment in any detail or carried out extensive empirical



	Hbb1_Salir	*Lgba_Phavu*
Hbb1_Salir	-	0.12
Hbb_human	0.14	0.07
Hbb_Horse	0.16	0.07
Hba_Human	0.11	0.07
Hba_Horse	0.11	0.07
Myg_Phyca	0.13	0.14
Glb5_Petma	0.16	0.12
Lgb2_Luplu	0.07	0.42
Lgba_Phavu	-	-

FIG. 4. Normalized weights for the addition of a trout β-globin (Hbb1_Salir) and a bean leghemoglobin (Lgba_Phavu) to the alignment of seven globin sequences in Fig. 1. The positions in the tree for the new sequences are shown with dashed lines. First, the trout β-globin is added, and the seven weights needed for this are shown. Then the bean globin is added, and eight weights are needed for this (one weight for each of the original seven globins and one for the trout globin). The weights have the effect of simultaneously upweighting sequences that are similar to the new sequence and downweighting sequences that have other close relatives in the tree. The weights are normalized to sum to 1.0.

evaluations. They are, however, intuitive and simple to calculate. It seems to us to be of great importance to make use of such weights or to develop and evaluate new weighting schemes as more and more databases of large alignments are created that need to be updated automatically. These weights help to exploit the information contained in large alignments. An alignment of a few hundred or more sequences that has been carefully created by experts, sometimes incorporating structural information, is an important and useful resource which contains much information to help characterize new sequences.

Phylogenetic Trees

All trees in CLUSTAL W are calculated using the neighbor-joining method.[18] These are used as guide trees to guide the multiple alignments or can be produced after multiple alignment with a bootstrap[34] option. This is a distance matrix approach which is fast to calculate but which gives the correct tree topology in a wide variety of situations. The method produces unrooted trees with estimates of branch lengths for each branch.

For the calculation of guide trees, the user can choose between fast approximate pairwise alignment of sequences using a *k*-tuple approach[35] based only on identities or full dynamic programming[36] with a weight matrix and two gap penalties. The former are extremely fast to calculate and are useful if the user has hundreds of sequences, but the latter are more accurate. Distances are calculated as the number of exact matches in the best alignment between the pair of sequences divided by the number of positions considered, ignoring positions with gaps. With the guide trees, there is no correction for multiple substitutions. The distances are given to the neighbor-joining method, and an unrooted guide tree is produced. The tree is rooted by a "midpoint" approach.[30] Root placement involves making the mean distance from the root to the tips of the tree equal on both sides of the root. The biological validity of this method of placing the root depends on the quality of the "molecular clock" in the data set. For the current application, however, it does not matter if the root is incorrectly placed. This is because placing the root using the midpoint method is the equivalent of always aligning the next most closely related sequences or groups of sequences on the tree, ending when all sequences are aligned. This is the desired behavior.

After multiple alignment (or after reading a full multiple alignment

[34] J. Felsenstein, *Evolution* **39,** 783 (1985).
[35] W. J. Wilbur and D. J. Lipman, *Proc. Natl. Acad. Sci. U.S.A.* **80,** 726 (1983).
[36] S. B. Needleman and C. Wunsch, *J. Mol. Biol.* **48,** 444 (1970).

from a file), more accurate trees can be calculated using distances calculated from the fully aligned sequences. For each distance calculation, gaps are not considered, but there is an option to ignore all sites where a gap occurs in any of the sequences. Although this sounds wasteful of data (it removes any site where a gap occurs) it does have the advantage of basing all distance calculations on the same number of sites. For sequences of similar length, this makes little difference. It has the added advantage of automatically removing the most ambiguous sites from the alignment (those that are hardest to align and where the exact alignment may be arbitrary or an artifact of the alignment process). There is a second option to correct the distances for multiple substitutions. This has little effect on small distances (e.g., sequences more than 80% identical) but will stretch large distances considerably to compensate for the great number of hidden substitutions that are estimated to have occurred. For example, using the Dayhoff model of protein evolution,[22] if one observes two amino acid sequences that are 20% identical (distance of 0.80 differences per site), one estimates that 250 substitutions have occurred per 100 sites (distance of 2.5 substitutions per site). Gaps are not considered in these distance calculations for two reasons. First, it is problematic how to score gaps, and, second, it is not known whether gaps appear and disappear in a clocklike manner. For many protein sequences, it appears that substitutions occur in a reasonably regular manner over time (the so-called molecular clock hypothesis), and this allows one to use sequences to derive phylogenetic information. There may not be an equivalent "gap clock," and if there is it is not known how to calibrate it.

To correct simple protein distances (number of observed differences per site) for multiple hits, it is common practice to use a formula from Kimura[37]:

$$K_{aa} = -\ln(1 - D - D^2/5) \tag{5}$$

where K_{aa} is the estimated, corrected distance and D is the observed distance. This formula is a curve-fitting approximation to a table of corrected distances derived from the Dayhoff model of protein evolution.[22] The formula is simple to calculate and is an extremely accurate fit to the Dayhoff model in the range 0.0 to 0.75 observed distance. Above this, however, the approximation becomes inaccurate, and at D above 0.85 or so, the formula cannot be evaluated as it requires finding the logarithm of a negative number. With CLUSTAL W, we use the Kimura formula for D up to 0.75 and use precalculated tables of corrected distances for all D above this in intervals of 0.001. These tables were calculated using the Dayhoff model

[37] M. Kimura, "The Neutral Theory of Molecular Evolution." Cambridge Univ. Press, Cambridge, 1983.

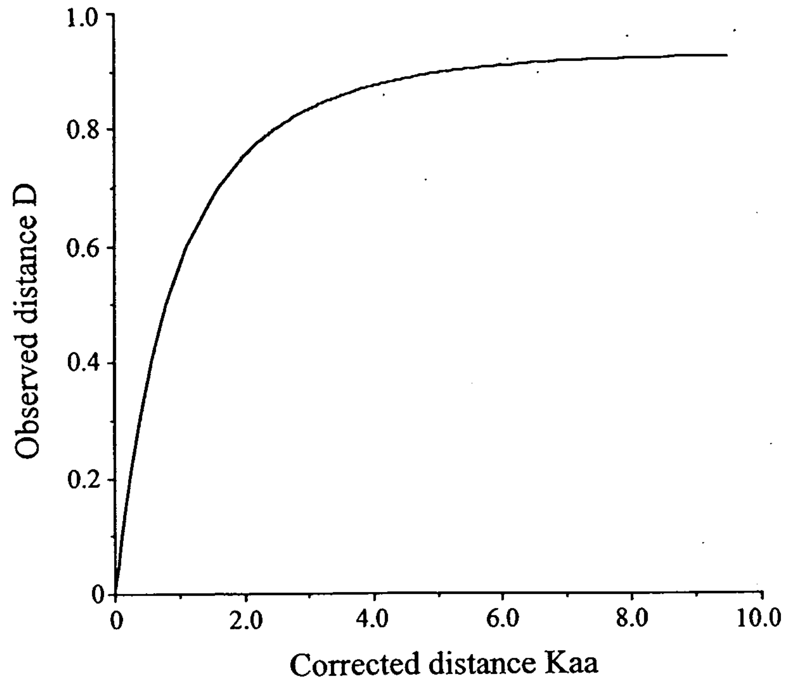


FIG. 5. Relationship between the simple observed distance between two protein sequences (mean number of differences per site) and the estimated corrected distance (the number of substitutions per site that have occurred, including multiple hits). These values were calculated using the Dayhoff model of amino acid substitution[22] and are hard coded in CLUSTAL W.

of amino acid substitution, from which the well-known PAM tables of amino acid similarity are derived. The graph in Fig. 5 shows the relationship between K_{aa} and D. These values were calculated using the Dayhoff model and hard coded into CLUSTAL W.

The Dayhoff model cannot correct any observed distances greater than 0.93 or so as the model reaches equilibrium at this point. For idealized proteins that have reached a divergence of 0.93 observed differences per site, substitutions are just as likely to restore an identity at a site as remove one. Therefore, we correct any observed distances above 0.93 to the arbitrary level of 10.0. This is crude, but such distances are rare with real sequences. If the sequences are this divergent, then alignment is very difficult to begin with and any tree making methods will have great problems making sensible trees. During bootstrapping, however, such distances can occur randomly, even if no sequences are as diverged as this. With CLUSTAL W, the user is warned if this occurs. Provided it occurs rarely, it seems to have no detrimental effect on the bootstrap results. If it becomes critical, however, users are advised to use the more sophisticated distance calculation method provided in the PROTDIST program of the PHYLIP package.[38] This method uses a Dayhoff model to estimate numbers of substitutions per site, but does so taking all mismatches into account as well as identities, and will be appropriate for sequences of any amino acid composition.

For graphical display of trees, we recommend that users use the DRAWGRAM and DRAWTREE programs of the PHYLIP package[38] or TREETOOL,[39] if the user has access to a SUN computer. For Macintosh computers, we provide the extremely useful and simple to use NJPLOT program of Manolo Gouy.[40] NJPLOT reads trees in the widely used New Hampshire format and allows the user to make simple manipulations and save the tree as PICT format, which can then be used in many Macintosh drawing and graphics packages. The bootstrap trees produced by CLUSTAL W include the bootstrap support levels as extra labels. These are displayed by both TREETOOL and NJPLOT.

Summary

We have tested CLUSTAL W in a wide variety of situations, and it is capable of handling some very difficult protein alignment problems. If the data set consists of enough closely related sequences so that the first alignments are accurate, then CLUSTAL W will usually find an alignment that is very close to ideal. Problems can still occur if the data set includes sequences of greatly different lengths or if some sequences include long regions that are impossible to align with the rest of the data set. Trying to balance the need for long insertions and deletions in some alignments with the need to avoid them in others is still a problem. The default values for our parameters were tested empirically using test cases of sets of globular proteins where some information as to the correct alignment was available. The parameter values may not be very appropriate with nonglobular proteins.

We have argued that using one weight matrix and two gap penalties is too simplistic to be of general use in the most difficult cases. We have replaced these parameters with a large number of new parameters designed primarily to help encourage gaps in loop regions. Although these new parameters are largely heuristic in nature, they perform surprisingly well and are simple to implement. The underlying speed of the progressive

[38] J. Felsenstein, *Cladistics* 5, 164 (1989).

[39] M. Maciukenas, University of Illinois, USA, unpublished. Available by anonymous ftp from rdp.life.uiuc.edu (/pub/RDP/programs/TreeTool).

[40] M. Gouy, University of Lyon, France, unpublished. Available for Apple Macintosh computers by anonymous ftp frpm ftp.ebi.ac.uk (/pub/software/mac/NJplot.sea.hqx).

alignment approach is not adversely affected. The disadvantage is that the parameter space is now huge; the number of possible combinations of parameters is more than can easily be examined by hand. We justify this by asking the user to treat CLUSTAL W as a data exploration tool rather than as a definitive analysis method. It is not sensible to automatically derive multiple alignments and to trust particular algorithms as being capable of always getting the correct answer. One must examine the alignments closely, especially in conjunction with the underlying phylogenetic tree (or estimate of it) and try varying some of the parameters. Outliers (sequences that have no close relatives) should be aligned carefully, as should fragments of sequences. The program will automatically delay the alignment of any sequences that are less than 40% identical to any others until all other sequences are aligned, but this can be set from a menu by the user. It may be useful to build up an alignment of closely related sequences first and to then add in the more distant relatives one at a time or in batches, using the profile alignments and weighting scheme described earlier and perhaps using a variety of parameter settings.

We give one example using SH2 domains. SH2 domains are widespread in eukaryotic signalling proteins where they function in the recognition of phosphotyrosine-containing peptides.[41] In the chapter by Bork and Gibson ([11], this volume), Blast and pattern/profile searches were used to extract the set of known SH2 domains and to search for new members. (Profiles used in database searches are conceptually very similar to the profiles used in CLUSTAL W: see the chapters [11] and [13] for profile search methods.) The profile searches detected SH2 domains in the JAK family of protein tyrosine kinases,[42] which were thought not to contain SH2 domains. Although the JAK family SH2 domains are rather divergent, they have the necessary core structural residues as well as the critical positively charged residue that binds phosphotyrosine, leaving no doubt that they are bona fide SH2 domains.

The five new JAK family SH2 domains were added sequentially to the existing alignment of 65 SH2 domains using the CLUSTAL W profile alignment option. Figure 6 shows part of the resulting alignment. Despite their divergent sequences, the new SH2 domains have been aligned nearly perfectly with the old set. No insertions were placed in the original SH2 domains. In this example, the profile alignment procedure has produced better results than a one-step full alignment of all 70 SH2 domains, and in

[41] I. Sadowski, J. C. Stone, and T. Pawson, *Mol. Cell. Biol.* **6,** 4396 (1986).
[42] A. F. Wilks, A. G. Harpur, R. R. Kurban, S. J. Ralph, G. Zuercher, and A. Ziemiecki, *Mol. Cell. Biol.* **11,** 2057 (1991).

```
CLUSTAL W(1.5) multiple sequence alignment


H_Src               EWYFGKI-----TRRESERLLLNA----ENPRGTFLVRESET---TKGAYCLSVSDFD--
Ce_B0523.1          AYFHGLI-----QREDVFQLLDN--------NGDYVVRLSDPKPGEPRSYILSVMFNN--
H_Crk               SWYWGRL-----SRQEAVALLQG------QRHGVFLVRDSST---SPGDYVLSVSENS--
H_SLP_76            EWYVSYI-----TRPEAEAALRK-----INQDGTFLVRDSSK-KTTTNPYVLMVLYKD--
M_3bp2              SVFVNTT-----ESCEVERLFKATDPRGEPQDGLYCIRNSST---KSGKVLVVWDESS--
H_PlcG1/1           KWFHGKLGAGRDGRHIAERLLTEYCIETGAPDGSFLVRESET---FVGDYTLSFWRNG--
H_PlcG1/2           EWYHASL-----TRAQAEHMLMR-----VPRDGAFLVRKRN----EPNSYAISFRAEG--
H_Ptp1c/1           RWFHRDL-----SGLDAETLLKG-----RGVHGSFLARPSRK---NQGDFSLSVRVGD--
H_Ptp1c/2           RWYHGHM-----SGGQAETLLQA-----KGEPWTFLVRESLS---QPGDFVLSVLSDQ--
Gg_Tensin           YWYKPDI-----SREQAIALLKD------REPGAFIIRDSHS---FRGAYGLAMKVASPP
H_Jak1              NGCHGPIC----TEYAINKLRQE-----GSEEGMYVLRWSCT-DFDNILMTVTCFEKS--
H_Tyk2              DGIHGPL-----LEPFVQAKLRP-------EDGLYLIHWSTS---HPYRLILTVAQRS--
R_Jak2              SNCHGPI-----SMDFAISKLKKAG----NQTGLYVLRCSPK---DFNKYFLTFAVER--
R_Jak3              ELCHGPI-----TLDFAIHKLKAAG----SLPGSYILRRSPQ---DYDSFLLTACVQTPL
Dm_Hop              LHCHGPI-----GGAYSLMKLHEN----GDKCGSYIVRECDR---EYNIYYIDINTKIMA
                                                    .  .              .

H_Src               ----------NAKGLNVKHYKIRKLD-----------------SGGFYITS----RTQFN
Ce_B0523.1          ---------KLDENSSVKHFVINSVE------------------NKYFVNN----NMSFN
H_Crk               ---------------RVSHYIINSSGPRPPVPPSPAQPPPGVSPSRLRIGD-----QEFD
H_SLP_76            ---------------KVYNIQIRYQK----------------ESQVYLLGTGLRGKEDFL
M_3bp2              --------------NKVRNYRIFEKD------------------SKFYLEG----EVLFA
H_PlcG1/1           ---------------KVQHCRIHSRQ--------------DAGTPKFFLTD----NLVFD
H_PlcG1/2           ---------------KIKHCRVQQEG------------------QTVMLGN-----SEFD
H_Ptp1c/1           ---------------QVTHIRIQNSG------------------DFYDLYG----GEKFA
H_Ptp1c/2           ------PKAGPGSPLRVTHIKVMCEG------------------GRYTVGG----LETFD
Gg_Tensin           PTVMQQNKKGDITNELVRHFLIETSP------------------RGVKLKG-CPNEPNFG
H_Jak1              -------EQVQGAQKQFKNFQIEVQK------------------GRYSLHG---SDRSFP
H_Tyk2              ------QAPDGMQSLRLRKFPIEQQD------------------GAFVLEG---WGRSFP
R_Jak2              -----------ENVIEYKHCLITKNE-----------------NGEYNLSG---TKRNFS
R_Jak3              G-------------PDYKGCLIRQDP-----------------SGAFSLVG---LSQLHR
Dm_Hop              KKTD-------QERCKTETFRIVRKD----------------SQWKLSYNN---GEHVLN
                                      .

H_Src               SLQQLVAYYSKH
Ce_B0523.1          TIQQMLSHYQKS
H_Crk               SLPALLEFYKIH
H_SLP_76            SVSDIIDYFRKM
M_3bp2              SVGSMVEHYHTH
H_PlcG1/1           SLYDLITHYQQV
H_PlcG1/2           SLVDLISYYEKH
H_Ptp1c/1           TLTELVEYYTQQ
H_Ptp1c/2           SLTDLVEHFKKT
Gg_Tensin           CLSALVYQHSIM
H_Jak1              SLGDLMSHLKKQ
H_Tyk2              SVRELGAALQGC
R_Jak2              SLKDLLNCYQME
R_Jak3              SLQELLTACWHS
Dm_Hop              SLHEVAHIIQAD
                     . .  .
```

Representative pre-aligned SH2 domains

JAK SH2 domains

FIG. 6. Profile alignment adding sequentially five newly detected JAK SH2 domains to an existing SH2 alignment. Because of space restrictions, just 10 of the 65 original SH2 domains are shown. All 65 domains were actually used for the profile alignment. The sole error occurs in H_Jak1 block 2 (shown in boldface italics) which is misaligned one residue leftward.

considerably less time. In this example, it is roughly five times faster to add the new sequences one at a time to the existing SH2 alignment than it is to recalculate the full alignment. It is also more accurate and gives the user greater control.

[23] Combined DNA and Protein Alignment

By JOTUN HEIN and JENS STØVLBÆK

Introduction

Most long DNA sequences contain coding regions, and thus it is optimal to use information from both the DNA sequence and the coded protein when comparing such a sequence to a homologous variant. In this chapter we present a heuristic algorithm that can compare DNA with both coding and noncoding regions, but also multiple reading frames, and determine which exons are homologous. A program, GenAl (genomic alignment), has been developed that implements the algorithm. It is demonstrated by comparing HIV2 (human immunodeficiency virus type 2) with HIV1. A stochastic model of the evolution of the complete virus has also been developed that allows estimation of the amount of selective constraint on different genes (including overlapping regions), the equilibrium base composition, and lastly the transversion and transition distances. This method has been applied to two HIV2's.

Comparisons of longer genomic DNA sequences will typically contain both coding and noncoding regions, which cannot be analyzed by the traditional dynamical programming algorithm.[1] As a protein evolves slower than its coding DNA, it will be more reliable to align the protein than the underlying DNA, and an algorithm that compares genomic DNA should incorporate the information from the protein. Presently, this problem is solved by separating the sequences into coding and noncoding parts, then analyzing them separately, and, finally, patching the resulting alignments into a global alignment. This is laborious and cannot be done if the DNA has overlapping reading frames. In this chapter we present an algorithm that solves these problems.

It should be noted that all evolutionary events happen at the DNA level, as proteins do not replicate. The basic events are (1) substitutions, which in coding regions can have the additional consequence of changing

[1] S. B. Needlemann and C. D. Wunsch, *J. Mol. Biol.* **48,** 444 (1970).

the coded protein; and (2) insertions–deletions (indels) in coding and noncoding regions. In coding regions, indels will normally have lengths that are a multiple of 3. Other evolutionary events are possible but are not allowed in a standard alignment.

Heuristic DNA–Protein Alignment Algorithm

Peltola *et al.*[2] introduced an algorithm that allowed a long DNA string to be searched for reading frames coding for a given protein sequence, that is, one protein versus one DNA string. Hein[3] considered algorithms aligning DNA sequences, but measuring the distances in terms of the induced change at the protein level. These algorithms were slow. The algorithm presented here is a fast but heuristic algorithm solving the same problem and is an elaboration on the algorithm first presented in Hein and Støvlbæk.[4]

The central approximation in the heuristic algorithm is to represent the coding DNA as if only the middle nucleotide in a codon coded for an amino acid. This allows a heuristic algorithm that is very similar to the simple DNA (or protein, but not both) comparing algorithm. This representation will create a new string with the same length as the original DNA string, but a position can now contain not only a nucleotide, but also an amino acid (potentially two amino acids, if there were a reading frame in the opposite direction). If two matched reading frames code for homologous proteins, there must have been a reading frame in the DNA all the way back to the most recent common ancestor and all indels should have lengths that are multiples of three nucleotides. In regions where no homologous proteins are matched, indels can have any length, as there could have been a period in its recent history when the DNA was noncoding. If the homology relationships between proteins in the two sequences are known beforehand, this could be part of the data and help in solving the problem. This option in the program is called guiding and is advantageous if the method could deduce the homology relationships itself. Because homologous proteins are usually more similar (less distant), there should be an inherent tendency to match homologous instead of nonhomologous proteins.

A general algorithm for aligning DNA with reading frames can now be formulated as follows: Let $s1$ and $s2$ be two sequences of length l_1 and l_2, respectively. The substring consisting of the first i elements of sk ($k = 1$, 2) is denoted sk_i, and $sk[i]$ refers to the ith element of sk. Let $D_{i,j}$ be the minimal distance between $s1_i$ and $s2_j$, when all insertion–deletions have

[2] H. Peltola, H. Söderlund, and E. Ukkonen, *Nucleic Acids Res.* **14,** 1.99 (1986).
[3] J. J. Hein, *J. Theor. Biol.* **167,** 169 (1994).
[4] J. J. Hein and J. Støvlbæk, *J. Mol. Evol.* **38,** 310 (1994).

lengths that are multiples of 3 and the differences in DNA are weighted by the amino acids that they invoke. Let the indices i' and j' be such that $i - i'$ (or $j - j'$) is 3, when only indels of length three are allowed, otherwise the value is 1. The distance between two elements in the sequences is $d[s1(i), s2(j)] = dn\{\text{nuc}[s1(i)], \text{nuc}[s2(j)]\} + da\{\text{aa}[s1(i)], \text{aa}[s2(j)]\}$, where $dn(\ ,\ )$ is the distance between two nucleotides (nuc) and $da\{\text{aa}[s1(i)], \text{aa}(s2)\}$ is the distance between the amino acids (aa) associated with that nucleotide. If there is one amino acid at each nucleotide, then it is the traditional amino acid distance; if there is only one amino acid as opposed to no amino acid, then it will be g_a, the cost of deleting one amino acid (g_n is the cost of deleting one nucleotide). In contrast to ordinary alignment, the matching term can also contain a weight that corresponds to the insertion–deletion of amino acids. The algorithm can then be written as

Initialization, $D_{0,j}$ (analogous for $D_{i,0}$):

$$D_{0,0} = 0$$
$$D_{0,j} = D_{0,j-1} + d[-, s2(j)]$$
$$D_{i,0} = D_{i-1,0} + d[s1(i), -]$$

The $D_{i,j}$ values then obey the recursion:

$D_{i,j}$ = minimum of following three quantities, when $i > 0$ and $j > 0$

1. Insertion in $s1$: $\{D_{i',j} + g_a{*}[\text{number of aa with codons overlapping } (i, i') + g_n]\}$
2. Insertion in $s2$: $\{D_{i,j'} + g_a{*}[\text{number of aa with codons overlapping } (j, j') + g_n]\}$
3. Matching nucleotides: $\{D_{i-1,j-1} + d[s1(i), s2(j)]\}$

The algorithm needs two sets of parameters, one set for proteins and one for DNA. Each set has a gap penalty and a distance function on the elements of the sequences: amino acids in the case of proteins and nucleotides in the case of DNA. The guiding principle for the parameters to be used is that very frequent events should have a low weight, whereas relatively rare events should have a higher weight. Many schemes have been devised to accomplish this.[5] A new problem in this analysis is how to weight events at the protein level relative to events at the DNA level. As proteins are more conserved than DNA and can retain observable similarity over much longer periods of time, the protein events should have higher costs than the DNA events. In principle it would be possible to let the protein level have complete precedence over the DNA level, corresponding to weighting proteins infinitely higher than DNA. DNA would still be a determining

[5] R. F. Doolittle, "Of URFs and ORFs." University Science Books, Mill Valley, California, 1986.

factor in the alignment, deciding the precise position of indels within a codon and choosing between equally good protein alignments. This algorithm can also handle overlapping reading frames.

The use of this algorithm is illustrated in two very simple cases in Fig. 1. Figure 2 shows the cost of different alignments, assuming different configurations of reading frames. It illustrates that the cheapest alignment for a given configuration of reading frames is the one that matches those reading frames. In general, it is more expensive to assume the presence of reading frames than their absence, because this avoids the cost associated with events for the corresponding reading frames. The information about the reading frames is essential to the algorithm, and it does not seem easy to modify the algorithm to also find the reading frames.

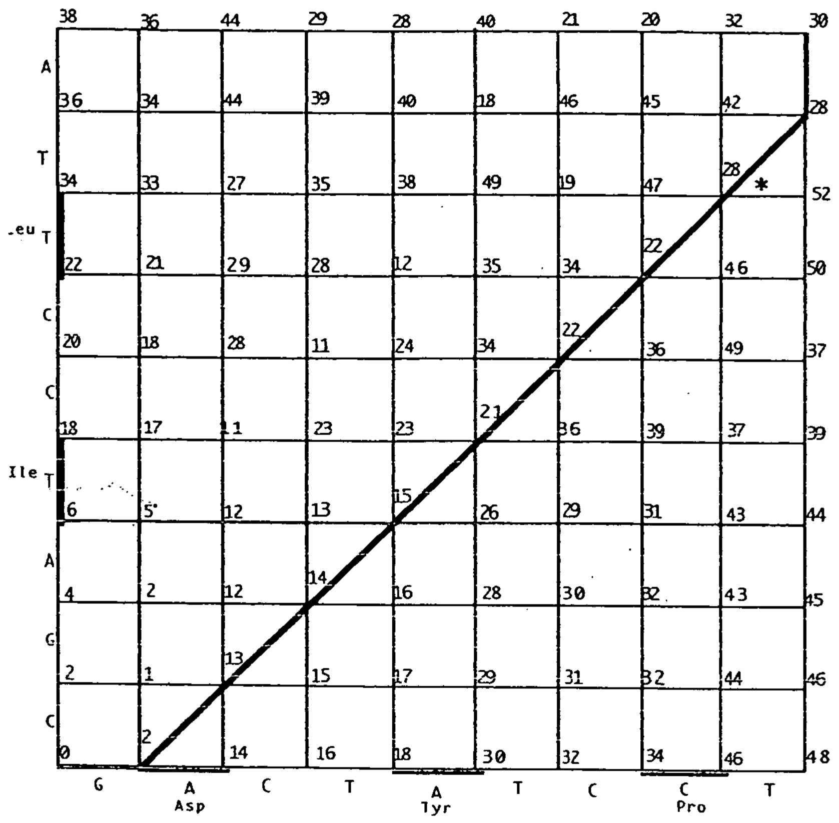
A program has been written, GenAl, that implemented this algorithm. GenAl can be obtained from the Netserver at EMBL. The algorithm was extended in several relevant ways that enhances its applicability to real data. (1) Frameshift indels do occur, but at very low frequency. This can be incorporated by allowing indels of length 1 and 2, but assigning them a very high cost. (2) Early in the history of sequence analysis it was realized that long stretches of gap signs in an alignment were likely to be due to one long indel and not to a series of adjacent small indels. This should be reflected in the gap cost function. The most used gap penalty function has the form $a + bk$ (where k is the length of the indel), as this allows for a fast algorithm).[6] This has been incorporated into GenAl and will improve the alignment involving medium size indels (<100 bp). For much larger indels a concave weighting would be superior).[7] (3) The program uses the Hirschberg linear memory algorithm,[8] so it can analyze very long sequences. (4) Alignment algorithms can be sped up and memory requirements reduced, without serious loss in reliability, by finding long stretches of common segments to the two sequences. This was implemented using the DAWG (directed acyclic word graph) as defined by Blumer *et al.*[9] This will also allow terminal indels to have a weight of zero, which is realistic if the sequences being compared have been sequenced to different extents. (5) The GenAl program can also search for close-to-optimum alignments, which can be useful in determining which regions are well aligned. Well-aligned regions will also have close-to-optimum alignments, while arbitrarily aligned regions will not.

[6] O. Gotoh, *J. Mol. Biol.* **162,** 705 (1981).

[7] W. Miller and E. W. Myers, *Bull. Math. Biol.* **50,** 2.97 (1988).

[8] D. S. Hirschberg, *Commun. ACM* **18,** 341 (1975).

[9] A. Blumer, J. Blumer, D. Hausler, and M. McConnell, *J. Assoc. Comput. Mach.* **34,** 3.578 (1987).

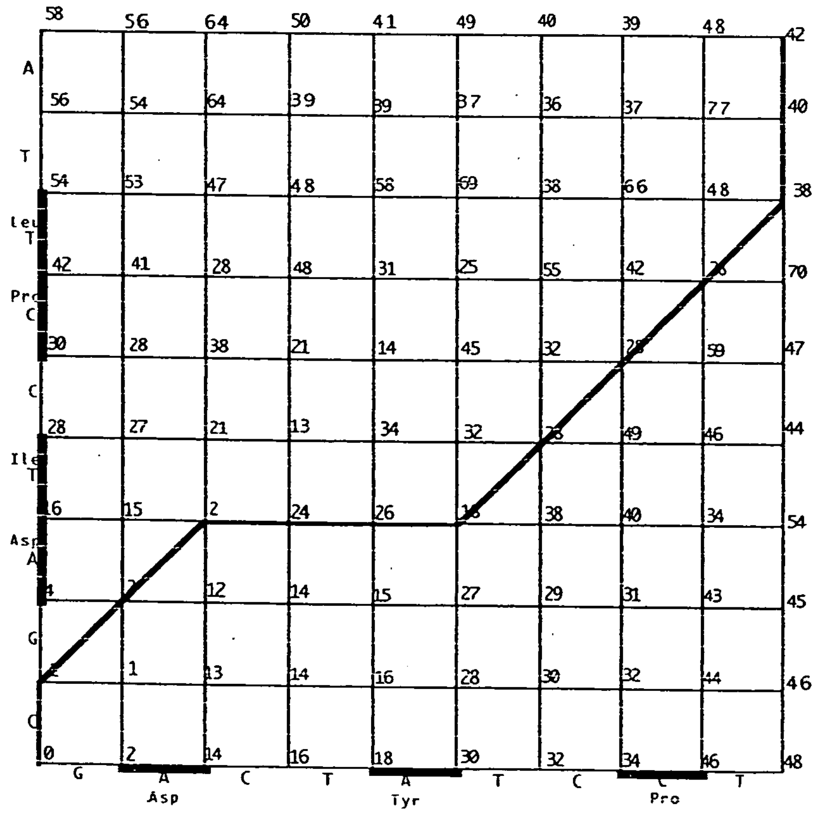


Total Alignment

```
  Asp  Tyr  Pro
G A C T A T C C T -
- C G A T C C T T A
   -    Ile   Leu
```

Protein Alignment

```
Asp Tyr Pro
 -  Ile Leu
```



Total Alignment

```
  Asp   Tyr   Pro
- G A C T A T C C T - -
C G A - - - T C C T T A
  Asp         Pro
          Ile    Leu
```

Protein Alignment

```
Asp   Tyr   Pro
Asp    -    Pro
     Ile    Leu
```

		Configuration		
		0-0	1-1	1-2
	0-0	$\underline{8}$	58 (8+50)	78 (8+70)
Alignment	1-1	11	$\underline{30}$ (11+19)	50 (11+39)
	1-2	12	62 (12+50)	$\underline{42}$ (12+30)

FIG. 2. If the two sequences shown in Fig. 1 were aligned without reading frames, the optimal alignment would be gapless and have a cost of 8. So now there are three different alignments, depending on which reading frames are assigned to the DNA sequences. The cost of the gapless alignment would be 58 if one reading frame were present in each sequence and 78 if the second sequence had two reading frames. The cheapest alignment (underlined) for a given configuration is the alignment that uses the information from the reading frames of that configuration.

FIG. 1. Alignment path, HIV2ST vs HIVJRFL. Two DNA sequences, with one reading frame each, are aligned in the upper matrix. The first sequence is at the x axis of the matrix and the second at the y axis. The weights used include (1) substitutions between nucleotides is 1, (2) a mutation of an amino acid is 5, (3) an indel of a nucleotide is 2, and (4) an indel of an amino acid is 10. In other words, events at the protein level cost five times the events at the DNA level. However, a protein indel within a singly coding region corresponds to three nucleotide indels. The distance between $s1_i$ and $s2_j$ is found at the node (i, j). $d(s1i, \text{-})$ (the cost of deleting the first i elements of $s1$) is $10 \times$ (number of amino acids in first i elements) $+ 2i$. The path up through the matrix corresponding to a minimal alignment is shown in thick lines. The total weight of the alignment is 30, which corresponds to the deletion of two nucleotides (cost 4), deletion of amino acids (cost 10), the cost of two amino acid replacements (cost 10), and last the substitution of six nucleotides. To calculate the value of the node (8,7) with one asterisk and value 28, three previous cells must be considered: (7,6) and, since (8,7) is within both reading frames, indels must come in groups of three and the relevant nodes will be (5,7) and (8,4). The value coming from (5,7) is 49 plus the cost for deleting three nucleotides (6) and one amino acid (10), which will be 65. The value coming from (8,4) is calculated analogously to be 53. The value coming from (7,6) is 22 plus the cost for both mutating a nucleotide (1) and an amino acid (5), which is 28. Since 28 is the smallest of the three, this value is assigned to (8,7). If a node is not within exons in both sequences, then indels do not have to come in groups of three. To calculate the value associated with node (5,2), which is outside the reading frame in sequence 2, the nodes that must be considered are (4,1), (5,1), and (4,2). The resulting total alignment and protein alignment is shown to the right of the matrix.

In the lower matrix an additional reading frame has been introduced into $s2$, so the algorithm has to choose which reading frame the protein in $s1$ is to be matched with in $s2$. It chooses the new reading frame and deletes the old reading frame. The cost of the alignment found is 42, which corresponds to the deletion of six nucleotides (12), the deletion of one reading frame of two amino acids (20), and the deletion of one additional amino acid (10).

```
                                      -----pol---------pol---------pol-----
                          -----gag---------gag---------gag-----
   1         303         548        1717        1840        2113        2122
  +-----------+-----------+-----------+-----------+-----------+-----------+
   0           1         113        1279        1402        1600        1609
                          -----gag---------gag---------gag---------gag-----
                                                  -----pol---------pol-----



  -----pol---------pol---------pol-----           -----vpx---------vpx-----
              -----vif---------vif---------vif---------vif-----
2122        4868        4930        4938        5343        5515        5552
  +-----------+-----------+-----------+-----------+-----------+-----------+
1609        4358        4413        4425        4807        4864        4876
              -----vif---------vif---------vif---------vif---------vif-----
  -----pol---------pol-----



              -----vpr---------vpr---------vpr---------vpr---------vpr-----
  -----vpx-----
                                      -----tat2--------tat2--------tat2----
5552        5681        5756        5844        5958        5980        5998
  +-----------+-----------+-----------+-----------+-----------+-----------+
4876        4876        4936        5024        5147        5166        5214
  -----vif---------vif-----                       -----tat2--------tat2----

  -----vpr---------vpr---------vpr---------vpr---------vpr-----



                                                  -----env---------env-----
              -----rev2----
  -----tat2--------tat2----
5998        6070        6139        6148        6149        6210        8291
  +-----------+-----------+-----------+-----------+-----------+-----------+
5214        5286        5361        5378        5541        5623        7668
  -----tat2--------tat2----           -----vpu---------vpu-----
              -----rev2----
                                                  -----env---------env-----



  -----env---------env---------env---------env---------env-----
  -----rev3--------rev3----
  -----tat3----                       -----nef---------nef---------nef-----
8291        8387        8544        8562        8584        8728        8729
  +-----------+-----------+-----------+-----------+-----------+-----------+
7668        7758        7900        7918        7942        8084        8086
  -----tat3----
  -----rev3--------rev3--------rev3--------rev3----
  -----env---------env---------env---------env---------env-----



  -----nef-----
8729        9329        9541        9672
  +-----------+-----------+-----------+
8086        8736        8896        8896
  -----nef-----
```

FIG. 3. Compact overview of the total alignment of a variant of HIV2 (HIV2ST) with a variant of HIV1 (HIVJRFL). The reading frames above the center line, which shows the sequence position, belong to HIV2 and the ones below to HIV1. Homologous reading frames are shown equidistant from the sequence line.

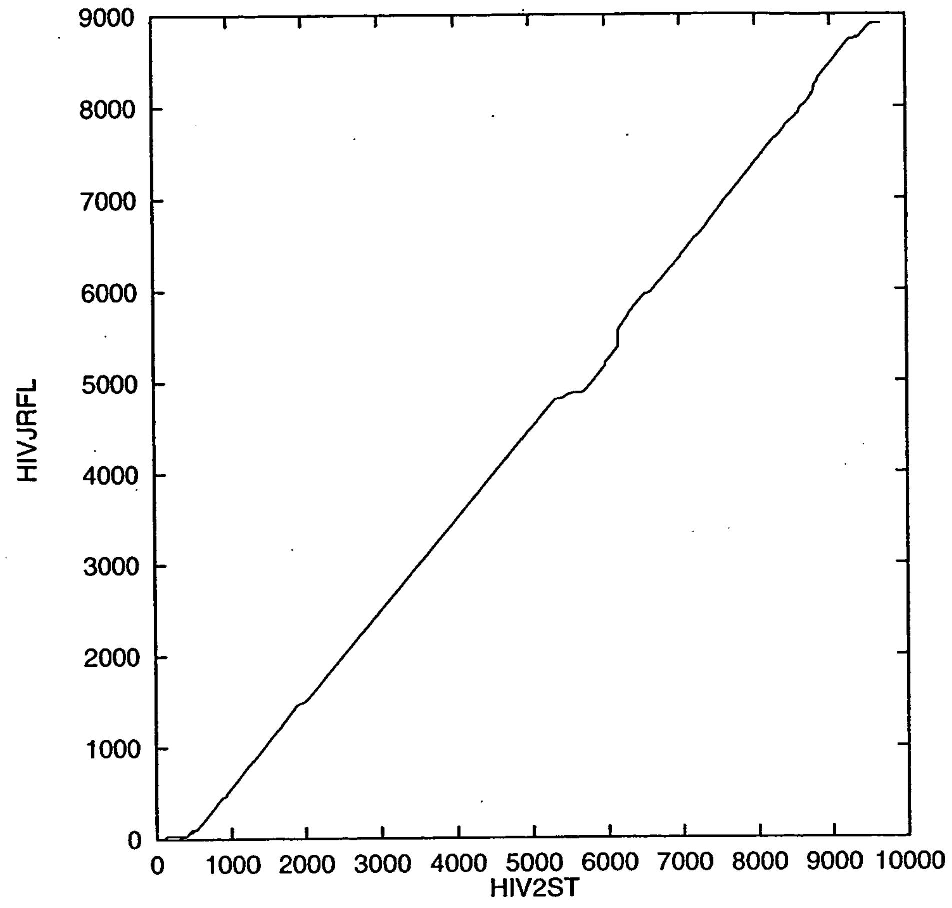


FIG. 4. Optimal alignment of two viruses (HIV2ST and HIVJRFL). The parameters used were 10 for the initiation of an indel, 1 for mutating a nucleotide, 5 for mutating an amino acid, and a nucleotide indel of length 1 cost 2 and an amino acid indel cost 10. Indels of length 1 or 2 in coding regions cost 80 (frameshifts). All these parameters are user options and could be changed.

Analysis of Human Immunodeficiency Virus

The input needed for GenAl is, then, the complete nucleotide sequences, a list of the genes in the two sequences, and the parameters used in the alignment. Each gene consists of a list of exons. An overview of the resulting alignment of HIV1 and HIV2 is shown in Fig. 3. Homologous exons are shown the same number of lines above and below the center line which represents the sequence. The optimal path up through the matrix is shown in Fig. 4. The HIV1 variant has not been fully sequenced at the ends.

Close to optimal alignments have been used to give a nonstatistical confidence estimate on the alignment by, for instance, Zuker.[10] If alignment paths that are within *d* of the optimum are also drawn, certain areas will

[10] M. Zuker, *J. Mol. Biol.* **221,** 403 (1991).

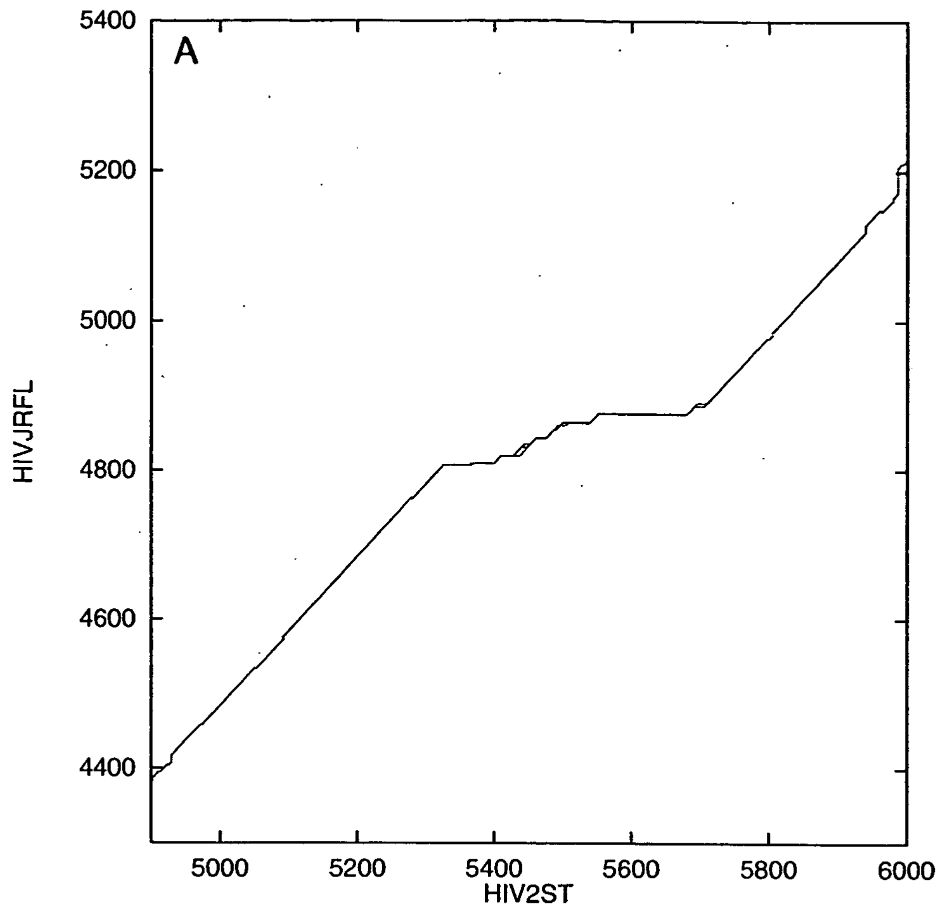


FIG. 5. Closeup of the alignment paths of HIV2 (in the region from the end of the *pol* gene to the end of the *vpr* gene) and HIV1 (in the region from the end of the *pol* gene to the beginning of the *vpu* gene). (A) Optimal paths. (B) Set of paths with a cost within 25 of the optimum for the same region. It can be seen, that the position of the large indel is badly determined.

contain very different paths, while other areas will only contain paths that are very similar to the optimal path. The first seems to be arbitrarily aligned, while the latter is aligned with greater certainty. Figure 5 gives an example of an optimal and a set of suboptimal (within 25 of the optimum) HIV2–HIV1 alignments. A small part of the total alignment, which is about 40 pages of text, is shown in Fig. 6.

Prospects

In longer sequences nonstandard events, like inversions, duplications, and transversions, are to be expected, and in such situations the present method will fail, as it can only interpret sequence evolution in terms of

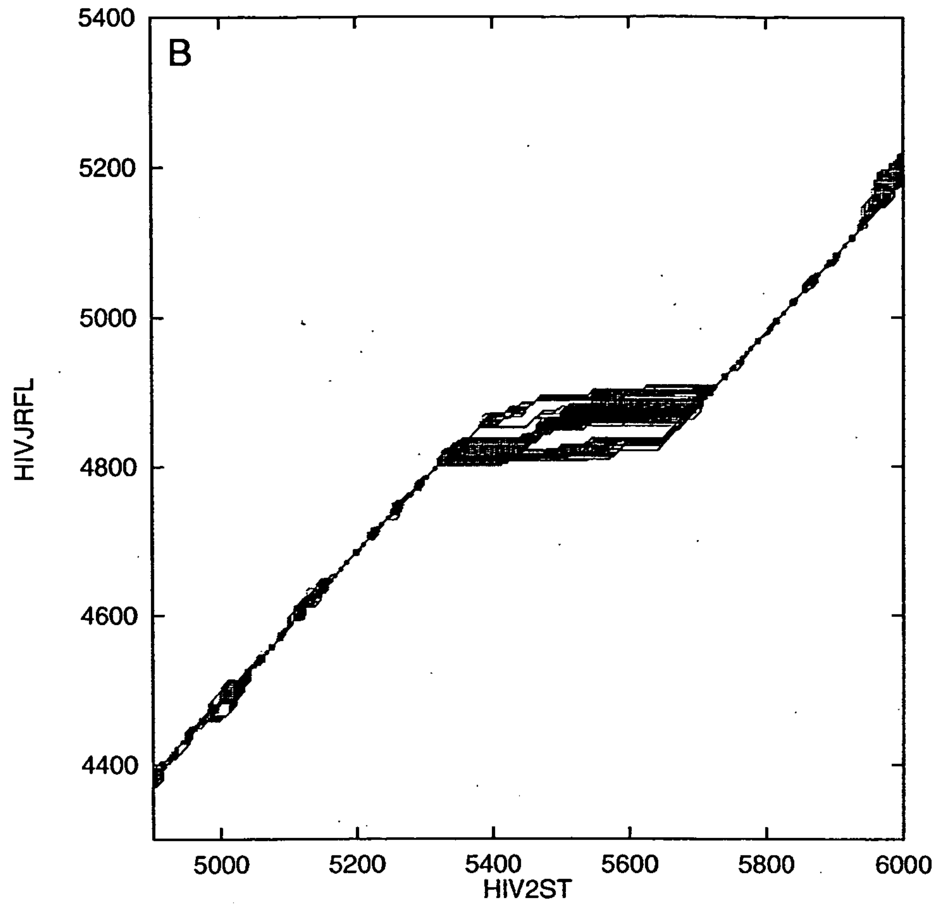


FIG. 5. (*continued*)

substitutions and indels. The present method could be combined with a method such as Schoeninger and Waterman[11] for analyzing such situations.

It would also be useful to have a multiple alignment version of this method. There are numerous methods that derive multiple alignments from pairwise alignments. However, the complexity of combining DNA and protein alignment, and the increased length of the sequences analyzed by this method, have so far prevented a satisfactory implementation of such a method. The presence of recombination in viruses would also undermine the rationale of using tree-based multiple alignment methods.

Evolutionary Model of Complete Virus

The events proposed by a minimal alignment of two homologous sequences give an underestimate of the amount of evolution that has occurred

[11] M. Schoeniger and M. S. Waterman, *Bull. Math. Biol.* **54,** 4.521 (1992).

```
      Asp   Arg   Gly   Leu   Pro   Ala   Ala   Arg   Glu   Thr   Arg   Asp
        Thr   Glu   Asp   Phe   Leu   Gln   Leu   Glu   Lys   Gln   Glu   Thr
2014 G A C A G A G G A C T T C C T G C A G C T C G A G A A A C A A G A G A C
1510 G A C A G C A A C T C C C T C T C A G - - - - - - A A G C A G G A G C C
        Thr   Ala   Thr   Pro   Ser   Gln               Lys   Gln   Glu   Pro
      Asp   Ser   Asn   Ser   Leu   Ser               Glu   Ala   Gly   Ala


      Thr   Met   Gln   Arg   Asp   Asp   Arg   Gly   Leu   Ala   Ala   Pro
        Pro   Cys   Arg   Glu   Thr   Thr   Glu   Asp   Leu   Leu   His   Leu
2050 A C C A T G C A G A G A G A C G A C A G A G G A C T T G C T G C A C C T
1540 G A T A G A C A A G G A A A T G T - - - A T C C T T T A A C T T C C C T
        Ile   Asp   Lys   Glu   Met         Tyr   Pro   Leu   Thr   Ser   Leu
      Asp   Arg   Gln   Gly   Asn   Val         Ser   Phe   Asn   Phe   Pro


      Gln   Phe   Ser   Leu   Trp   Lys   Arg   Pro   Val   Val   Thr   Ala
        Asn   Ser   Leu   Phe   Gly   Lys   Asp   Gln   !!!
2086 C A A T T C T C T C T T T G G A A A A G A C C A G T A G T C A C A G C A
1573 C A G A T C A C T C T T T G G C A A C G A C C C C T C G T C A C A A T A
        Arg   Ser   Leu   Phe   Gly   Asn   Asp   Pro   Ser   Ser   Gln   !!!
      Gln   Ile   Thr   Leu   Trp   Gln   Arg   Pro   Leu   Val   Thr   Ile
```

FIG. 6. Alignment showing part of the small region involved in the coding of both the *pol* gene and the end of the *gag* gene. The amino acids just above and below the DNA belong to the gag protein, and the uppermost and lowermost amino acids belong to the *pol* protein. There are two indels in this region, one of length 3 and one of length 6.

since their most recent common ancestor. To correct for this and to analyze the evolutionary dynamics, an evolutionary model is needed.

A model is presented for sequence evolution that can analyze combinations of noncoding, singly coding, and multiply coding regions of aligned homologous DNA sequences. It combines features of the Hasegawa, Kishino, and Yano (HKY) substitution model,[12] allowing for bias both in nucleotide frequencies and in transition/transversion rates with those of the Li *et al.*[13] transition–transversion model, with selection against replacement substitutions. In addition, it is generalized to apply to any combination of overlapping reading frames.

Hasegawa, Kishino, and Yano

The HKY model is a stationary continuous time Markov chain. The rate matrix has three parameters for the equilibrium base frequencies, π_A, π_C, and π_G [π_T is not a free parameter because $\pi_T = (1.0 - \pi_A - \pi_C - \pi_G)$] and two parameters for the transition rate, α, and the transversion rate, β. The transition probability will also have a time parameter, t. The

[12] M. Hasegawa, H. Kishino, and T. Yano, *J. Mol. Evol.* **22,** 160 (1985).
[13] W.-S. Li, C.-I. Wu, and C.-C. Luo, *Mol. Biol. Evol.* **22,** 150 (1985).

$$\begin{array}{ccccc}
Q_{i,j} & A & C & G & T \\
A & - & \beta*\pi C & \alpha*\pi G & \beta*\pi T \\
C & \beta*\pi A & - & \beta*\pi G & \alpha*\pi T \\
G & \alpha*\pi A & \beta*\pi C & - & \beta*\pi T \\
T & \beta*\pi A & \alpha*\pi C & \beta*\pi G & -
\end{array}$$

Equilibrium frequencies of nucleotides: $(\pi A, \pi C, \pi G, \pi T)$.

Transition Probabilities: $P(t) = e^{tQ}$

Distance between sequence: $2*t*[\beta*\pi Y*\pi R + \alpha*(\pi T*\pi C + \pi G*\pi A)]$

$\pi R = (\pi A + \pi G)$ (purines), $\pi Y = (\pi C + \pi T)$ (pyrimidines)

FIG. 7. Hasegawa, Kishino, and Yano substitution model. The rate matrix, **Q**, of the HKY model assumes a substitution rate toward a nucleotide (N) that is proportional to its equilibrium frequency, π_N, multiplied by the transition rate (α) if the substitution is a transition or by the transversion rate (β) if the substitution is a transversion. The element in a row has sum 0.0, so the diagonal entry (—) will be minus the sum of the off-diagonal elements in a row. The transition probability matrix, **P**, can be calculated by matrix exponentiation of $t\mathbf{Q}$ ($\mathbf{P} = e^{t\mathbf{Q}}$), where t is the amount of time elapsed. The entries of P have five parameters π_A, π_C, π_G, a, and b, where $a = \alpha t$ and $b = \beta t$. For instance, the probability for changing an A to a C can be written as $P_{A,C}(\pi_A, \pi_C, \pi_G, a, b)$. The expected number of events per site is used as a distance measure between the two sequences and is readily calculated from the parameters.

time parameter will always appear as αt and βt, and the rates and time cannot be estimated separately and are formulated as a distance. Figure 7 illustrates the rate matrix of this model.

Selection against Replacements

The analysis of sequences coding for proteins is complicated by selection against substitutions that cause amino acid replacements. For single coding sequences this causes substitutions to be partitioned into synonymous (also called silent) and replacement (also called nonsynonymous) substitutions.

Li *et al.*[13] modeled the evolution of coding regions by assuming that the genetic code was more regular than it actually is. It was assumed that varying any nucleotide in a codon among the four possible nucleotides would give the same amino acid (4-fold degenerate) (4), two amino acids, each pair differing by a transition (2-fold degenerate) (2 : 2), or four different amino acids (nondegenerate) (1 : 1 : 1 : 1). It was also assumed that the nucleotide at different positions evolved independently.

Each position is modeled by the two-parameter model of Kimura,[14] which is the special case of the HKY model where $\pi_T = \pi_A = \pi_C = \pi_G$. It is also assumed that any replacement substitution is accepted with a probability f (selection factor), expected for purifying selection to lie between 0.0 and 1.0. This can readily be incorporated into the scheme of Kimura by defining α' and β' for the three site types. In a nondegenerate site, both transitions and transversions will change the amino acid, and therefore we will have $\alpha' = f\alpha$ and $\beta' = f\beta$; in a 2-fold degenerate site only transversions will be amino acid changing, yielding $\alpha' = \alpha$ and $\beta' = f\beta$; and a 4-fold degenerate site will have no selection factor incorporated, with $\alpha' = \alpha$ and $\beta' = \beta$ (Fig. 8A).

Purifying selection will slow down all events in nondegenerate sites and only transversions in 2-fold degenerate sites; it should have no effect in 4-fold degenerate sites. Properly quantified, this slowdown can be used to estimate the strength of selection and then to recover the underlying biochemical substitution rate. It has been shown in many data sets that the HKY model is superior to the Kimura two-parameter model),[15] so this underlying model was used.

Generalization

This scheme is easily carried over to overlapping reading frames (Fig. 8B) and will be illustrated in the case of two overlapping reading frames. Now, there will be a selection factor for each protein, f_1 and f_2, and one for both $f_{1,2}$, and each site will have to be classified according to its kind in each reading frame. If selection works independently on the two genes, $f_{1,2} = f_1 f_2$.

For the model of the complete virus each gene has its own selection intensity that must be the same for different regions. The selection induced by different genes in areas of overlap are assumed to work independently. This gives a model with five parameters from the HKY model plus nine selection factors from nine genes. The parameters were then estimated by maximum likelihood.[16] Figure 9 shows the overview of an HIV1–HIV1 alignment, and the estimated parameters are shown in Fig. 10. The estimated distance is 50% larger in this model than a previous model by Hein

[14] M. Kimura, *J. Mol. Evol.* **16,** 111 (1980).
[15] N. Goldman, *J. Mol. Evol.* **36,** 182 (1993).
[16] A. W. F. Edwards, "Likelihood." Cambridge Univ. Press, Cambridge, 1972.

A

Sites types

1-1-1-1:	α' = f*α	β' = f*β
2-2 :	α' = α	β' = f*β
4 :	α' = α	β' = β

B

2nd RF\1st RF	1-1-1-1	2-2	4
1-1-1-1:	(f1*f2*α,f1*f2*β)	(f2*α,f1*f2*β)	(f2*α,f2*β)
2-2 :	(f1*α,f1*f2*β)	(α,f1*f2*β)	(α,f2*β)
4 :	(f1*α,f1*β)	(α,f1*β)	(α,β)

C

```
2. pol:     Arg  Gly
1. gag:       Glu  Asp
            A G A G G A
            A G C A A C
1. gag:       Ala  Thr
2. pol:     Ser  Asn
```

FIG. 8. (A) Single reading frame selection. With one reading frame all amino acid changing substitutions will be assumed to be reduced with a factor, f, relative to if they had not changed the amino acid. Ignoring a few irregularities of the genetic code allows this f to be incorporated directly into the Kimura model. (B) Two independent overlapping reading frame selection. If two reading frames, 1 and 1, are present, there will be two distinct selection factors, f_1 and f_2, that can be incorporated as well. If there were interaction between the joint effects of replacements in both reading frames, then $f_1 f_2$ should be replaced with $f_{1,2}$. This is readily generalized to more than two reading frames. (C) Likelihood for AC. Using the upper sequence to classify sites, then it is a 1-1-1-1 site with respect to the second reading frame and a 2-2 site with respect to the first reading frame. Thus, $a' = f_2 a$ and $b' = f_1 f_2 b$. The probability for observing AC is $pA{*}P_A, c(p_A, p_C, p_G, f_2 a, f_1 f_2 a)$, that is, the probability of picking A according to the equilibrium distribution and then having A evolve into C, subject to the selective slowdown from the two reading frames of which it is a part.

and Støvlbæk[17] that used the Kimura two-parameter model, not HKY, as underlying model.

As a statistical model for the evolution of the virus, the present model is lacking in biological realism. A more complete model should take the

[17] J. J. Hein and J. Støvlbæk, *J. Mol. Evol.* **40,** 181 (1995).

```
             -----gag---------gag-----           -----vif---------vif-----
                         -----pol---------pol---------pol-----
   1          336         1634        1838        4587        4642        5105
  +-----------+-----------+-----------+-----------+-----------+-----------+
   1          336         1631        1874        4623        4678        5141
                         -----pol---------pol---------pol-----
             -----gag---------gag-----           -----vif---------vif-----



 -----vpr---------vpr---------vpr-----
 -----vif-----                                   -----rev2----
                         -----tat2--------tat2--------tat2----
5105        5165        5376        5395        5515        5590        5607
  +-----------+-----------+-----------+-----------+-----------+-----------+
5141        5201        5412        5431        5551        5626        5643
                         -----tat2--------tat2--------tat2----
 -----vif-----                                   -----rev2----
 -----vpr---------vpr---------vpr-----



             -----env---------env---------env---------env---------env-----
                                     -----rev3--------rev3--------rev3----
 -----vpu---------vpu-----           -----tat3--------tat3----
5607        5770        5852        7915        7960        7999        8189
  +-----------+-----------+-----------+-----------+-----------+-----------+
5643        5803        5888        7972        8017        8056        8246
 -----vpu---------vpu-----           -----tat3----
                                     -----rev3--------rev3--------rev3----
             -----env---------env---------env---------env---------env-----



 -----env---------env-----
 -----rev3----
                                     -----nef-----
8189        8195        8331        8333        8953        8956        9175
  +-----------+-----------+-----------+-----------+-----------+-----------+
8246        8252        8388        8390        9007        9010        9229
                                     -----nef---------nef-----

 -----env---------env-----



9175        9176
  +-----------+
9229        9229
```

FIG. 9. GenAl was used to align two HIV1's (HIVELI and HIVLAI) with the same parameters as in Fig. 3.

following into account: varying rates at different positions, codon bias, and uneven selection along the gene and recombination. Recombination will have an effect because the time back until the most recent common ancestor for the two sequences can differ from region to region. More importantly,

```
Hasegawa, Kishino & Yano Substitution Model Parameters:

α*t      β*t        πA        πC       πG       πT

0.350    0.105      0.361     0.181    0.236    0.222

0.015    0.005      0.004     0.003    0.003


                    Selection Factors

                 GAG  0.385 (s.d. 0.030)
                 POL  0.220 (s.d. 0.017)
                 VIF  0.407 (s.d. 0.035)
                 VPR  0.494 (s.d. 0.044)
                 TAT  1.229 (s.d. 0.104)
                 REV  0.596 (s.d. 0.052)
                 VPU  0.902 (s.d. 0.079)
                 ENV  0.889 (s.d. 0.051)
                 NEF  0.928 (s.d. 0.073)


          Estimated Distance per Site: 0.194
```

FIG. 10. HIV1 analysis, showing parameters estimated for the model describing the evolution of the aligned HIV1's in Fig. 9. Transitions happen more than three times as frequently as transversions. The sequences also seem very A rich. All estimated parameters are accompanied by a standard deviation. It is seen that longer genes (*gag*, *pol*, *env*) have their selection factors better determined than shorter genes (i.e., they have smaller standard deviations). It is also seen that the *pol* gene is the slowest evolving. The *tat* gene has an *f* value higher than 1.0, and the three last small genes have *f* values that are very high. The large envelope gene (*env*) has evolved considerably faster than the other two large genes (*pol*, *gag*). The expected number of events per site is 0.195.

a model should allow not only the analysis of two sequences but of many, which means that the phylogeny with recombination problem would have to be addressed.

Conclusion

The basic idea is to combine the protein alignment problem with the DNA alignment problem and then solve them simultaneously. The alignment will then also align homologous exons with homologous exons, because this is more parsimonious.

The solution of the combined alignment problem is a mosaic of noncoding, singly coding, and possibly multiply coding regions. It is of interest to measure rates of evolution and selective effects of different coding regions and to estimate the expected number of events per site in the absence of selection as a distance measure between the sequences. A model has been proposed that combines one of the better models for the evolution of nucleotides with one of the most widely used methods of incorporating selection against amino acid replacements to yield a model for complete viruses, including regions with overlapping reading frames.

Acknowledgments

J. H. was supported by the Carlsberg Foundation and the Danish Research Council, Grants 11-8916-1 and 11-9639-1. J. S. was supported by the Danish Research Council, Grants 11-8916-1 and 11-9639-1, and by the Carlsberg Foundation. We thank Bernt Guldbrandtsen for comments on the manuscript and for help with the figures, and Dr. G. Myers for making the HIV database available.

[24] Inferring Phylogenies from Protein Sequences by Parsimony, Distance, and Likelihood Methods

By JOSEPH FELSENSTEIN

Introduction

The first molecular sequences available were protein sequences, so it is not surprising that the first papers on inferring phylogenies from molecular sequences described methods designed for proteins. Eck and Dayhoff[1] described the first molecular parsimony method, with amino acids as the character states. Fitch and Margoliash[2] initiated distance matrix phylogeny methods with analysis of cytochrome sequences. Neyman[3] presented the first likelihood method for molecular sequences, using a model of symmetric change among all amino acids.

After a long period in which attention shifted to nucleotide sequences, attention is again being paid to models in which the amino acid sequences explicitly appear. This is not only because of the increased availability of protein sequence data, but also because the conservation of amino acid sequence and protein structure allows us to bring more information to bear on ancient origins of lineages and of genes. In this chapter I briefly review

[1] R. V. Eck and M. O. Dayhoff, "Atlas of Protein Sequence and Structure 1966." National Biomedical Research Foundation, Silver Spring, Maryland, 1966.

[2] W. M. Fitch and E. Margoliash, *Science* **155,** 279 (1967).

[3] J. Neyman, *in* "Statistical Decision Theory and Related Topics" (S. S. Gupta and J. Yackel, eds.), p. 1. Academic Press, New York, 1971.

the work on using protein sequence and structure to infer phylogeny, in the process describing some methods of my own.

Parsimony

Eck and Dayhoff[1] did not describe their algorithms in enough detail to reproduce them, but it is apparent that the model of amino acid sequence evolution they used did not take the genetic code into account. It simply considered the amino acids as 20 states, with any change of state able to result in any of the other 19 amino acids. The realization that more information could be extracted by explicitly considering the code shortly led to more complex models. Fitch and Farris[4] gave an approximate algorithm to calculate for any set of amino acid sequences, on a given tree, how many nucleotide substitutions must, at a minimum, have occurred. As certain amino acid replacements would then require two or three base substitutions, this would differentially weight amino acid replacements. Moore[5,6] had already presented an exact, though more tedious, algorithm to count the minimum number of nucleotide substitutions needed, and he pointed out the approximate nature of the Fitch and Farris method.[7]

These papers might have settled the matter for the parsimony criterion, except that they count as equally serious those nucleotide substitutions that do and do not change the amino acid. For example, we might have a phenylalanine that is coded for by a UUU, which ultimately becomes a glutamine that is coded for by a CAA. This requires three nucleotide substitutions. It is possible for one of these to be silent, as we can go from UUU (Phe) $\rightarrow$ CUU (Leu) $\rightarrow$ CUA (Leu) $\rightarrow$ CAA (Glu). Presumably the second of these changes will not be as improbable as the others, as it will not have to occur in the face of natural selection against change in the amino acid, nor wait for a change of environment or genetic background that favors the amino acid replacement.

In the PROTPARS program of my PHYLIP package of phylogeny programs, I have introduced (in 1983) a parsimony method that attempts to reflect this. In the above sequence it counts only two changes, allowing the silent substitutions to take place without penalty. In effect the method uses the genetic code to designate which pairs of amino acids are adjacent, and it allows change only among adjacent states. Sankoff[8] and Sankoff and

[4] W. M. Fitch and J. S. Farris, *J. Mol. Evol.* **3,** 263 (1974).
[5] G. W. Moore, J. Barnabas, and M. Goodman, *J. Theor. Biol.* **38,** 459 (1973).
[6] G. W. Moore, *J. Theor. Biol.* **66,** 95 (1977).
[7] G. W. Moore, *in* "Genetic Distance" (J. F. Crow and C. Denniston, eds.), p. 105. Plenum, New York, 1974.
[8] D. Sankoff, *SIAM J. Appl. Math.* **28,** 35 (1975).

Rosseau[9] have presented a generalized parsimony algorithm that allows us to count on a given tree topology how many changes of state are necessary, where we can use an arbitrary matrix of penalties for changes from one state to another. The PROTPARS algorithm is equivalent to the Sankoff algorithm, being quicker but less general.

The set of possible amino acid states in the PROTPARS algorithm has 23 members, these being the 20 amino acids plus the possibilities of a gap and a stop codon. Serine is counted not as one amino acid but as two, corresponding to the two islands of serine codons in the genetic code. These are {UCA, UCG, UCC, UCU} and {AGU, AGC}, which make serine the only amino acid whose codons fall into two groups that cannot be reached from each other by a single mutation. PROTPARS copes with this by regarding them as two amino acid states (ser1 and ser2) and treats an observation of serine as an ambiguity between these two.

Imagine that we know, for a node in the tree, the set of amino acid states that are possible at this node. If the node is a terminal (tip) species, these are just the observed amino acid, there being more than one if serine is observed or if any of Asn, Gln, or Glx are observed. There is also the possibility that the amino acid is unknown, but known not to be a gap, and the possibility that the amino acid could be any one including a gap. More complex ambiguities are also possible and can arise in the process of reconstruction of the states at interior nodes in the tree. Any of these can be represented by designating the members of the set S_0 of possible states.

Given the particular version of the genetic code that we are using, we can also precompute, for each amino acid a, the set N_a of amino acid states that are one or fewer steps away. In PROTPARS, gaps are counted as being three steps away from all the amino acids and from stop codons. Having these precomputed sets allows us to take the sets S_0 at the tips of the tree, and compute for them S_1 and S_2, the sets of amino acid states one or fewer steps away, and two or fewer steps away. In our program, all states, including gaps, are three or fewer steps away, so that we do not need a set S_3. In PROTPARS the three sets S_0, S_1, and S_2 are updated down the tree, and the number of steps needed for the tree counted, in the following way.

Imagine that there is an internal node in the tree with two descendants, and whose sets of possible states are the L_i and the R_i. We are computing the sets S_i for the internal node. First, L_0 and R_0 are compared. If they are the same then the L_i must be identical to the R_i, and the S_i are simply

[9] D. Sankoff and P. Rousseau, *Math. Prog.* **9**, 240 (1975).

set to be the L_i, and no steps are counted. Otherwise, we compute the four sets

$$\begin{aligned} T_0 &= L_0 \cap R_0 \\ T_1 &= (L_1 \cap R_0) \cup (L_0 \cap R_1) \\ T_2 &= (L_2 \cap \mathrm{R}_0) \cup (L_1 \cap R_1) \cup (L_0 \cap R_2) \\ T_3 &= R_0 \cup (L_2 \cap R_1) \cup (L_1 \cap R_2) \cup L_0 \end{aligned} \tag{1}$$

They are computed one after the other. Their interpretation is straightforward. For example, T_1 is the set of amino acid states that, if present at the internal node, requires one step to give rise to L_0 and none to give rise to R_0, or else one step to give rise to R_0 and none to give rise to L_0. Thus it is the set of states which, if present at the interior node, require one extra step in the subtree that is above that node. As soon as one of these, say T_k, turns out to be nonempty, we know that a minimum of k more steps will be needed at this node, and that the set S_0 for that node will be T_k. In addition, T_3 at least must be nonempty, as it contains the union of R_0 and L_0.

Now, having found S_0, all we need to do is to compute S_1 and S_2 for the internal node. The formulas for doing so are

$$S_k = \bigcup_{a \in S_{k-1}} N_a, \qquad k = 1,2 \tag{2}$$

This of course does not need to be done if $L_0 = R_0$, as the sets L_1 and L_2 (or R_1 and R_2) can then be used directly.

This method of calculation using sets is equivalent to having a vector of numbers, one for each amino acid state, which are 0, 1, 2, or 3. The Sankoff algorithm asks us to specify for each state the number of extra steps that would be required above that point in the tree if that state existed in that internal node. In our model the possible values for the number of extra steps are 0, 1, 2, and 3. The sets S_i are just the amino acids that would have the number of extra steps less than or equal to i. The algorithm is then equivalent to the appropriate application of the Sankoff algorithm. It could probably be speeded up further, as most of the time the set S_2 is the set of all amino acids, and that could be used as the basis for some further economies.

Figure 1 shows the sets that would be stored on a small sample tree for one amino acid position, and the counting of steps. At each node the three sets S_0, S_1, and S_2 are shown, and at interior nodes the number of steps that are counted are also shown in circles. There are four different amino acids at the tips of the tree. If any amino acid could change to any other the tree would require only three steps, but in my protein parsimony model it requires five.

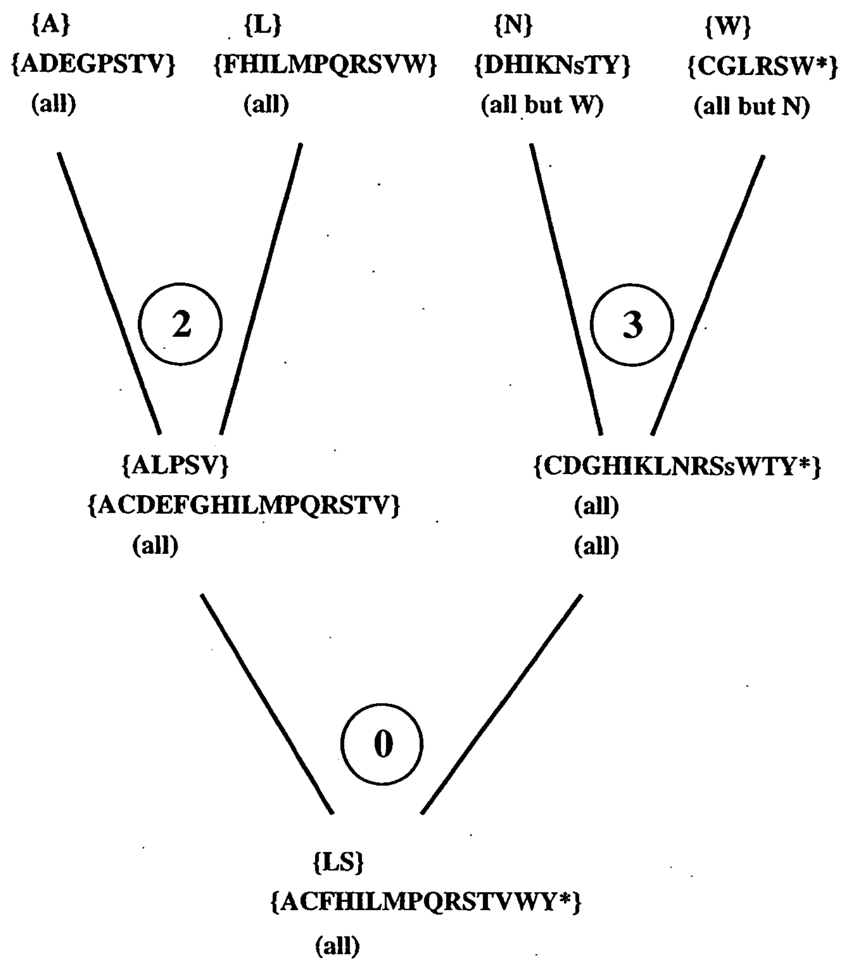


FIG. 1. Small tree with the calculation of the sets S_0, S_1, and S_2 shown at each node, for a site where the tips have amino acid states alanine, leucine, asparagine, and tryptophan, respectively. The sets are shown as sets of one-letter amino acid representations. S and s are the two codon "islands" of serine, and an asterisk (*) represents stop codons. The number of steps counted at each fork is shown in a circle.

Protein parsimony methods exactly equivalent to PROTPARS are also available in the programs PAUP and MacClade, using predefined matrices of costs of substitution between amino acid states, with the costs being taken into account by the Sankoff algorithm.

Distances

Distance matrix methods calculate for every pair of sequences an estimate of the branch length separating them, where branch length is the product of time and rate of evolution. That tree is then chosen that, by some criterion, makes the best prediction of these pairwise distances. For protein sequences we need to specify a probabilistic model of evolution. Jukes and Cantor[10] were the first to do this for protein sequences (see also Farris[11]). This model was highly oversimplified, as it had equal probabilities of change between all pairs of amino acids. Dayhoff and Eck[12] and Dayhoff *et al.*[13] empirically tabulated probabilities of change between amino acids over short evolutionary times, producing a table of transition probabilities between amino acids. This model does not take explicit account of the genetic code, and it is subject to errors from the limited sample size on which it was based. Nevertheless, the genetic code should affect its transition probabilities, and so should the biochemical properties of the amino acids. A more recent empirical model of amino acid change is that of Jones *et al.*[14] They have also produced models for specific subclasses of proteins that may be more useful in those contexts.[15] Other compilations of scoring matrices for evaluating the similarity of amino acid sequences[16,17] are not in the form of transition probability tables. For this reason they cannot be used to compute the branch length estimates that we require here.

A naive alternative to these empirical matrices is to divide the amino acids into a number of categories, based on their chemical properties. Suppose that we imagine mutations occurring in the genetic code table, with the starting points being codons generated at random from a given base composition. Now imagine single base substitutions. If these do not change the biochemical class of the amino acid, they are accepted; if they do, they are only accepted with probability p. We omit the stop codons from consideration: if either the starting point or the destination of a change is a stop codon, the change is not made. This model, once given the amino acid categories, the base frequencies, and the probability p, generates a transition probability table between all pairs of amino acids.

Version 3.5 of PHYLIP contains a program, PROTDIST, which computes distances based on the PAM001 model[13] and the transition probability matrix generated by the categories model. It also can compute distances

[10] T. H. Jukes and C. Cantor, *in* "Mammalian Protein Metabolism" (M. N. Munro, ed.), p. 21. Academic Press, New York, 1969.
[11] J. S. Farris, *Am. Nat.* **107,** 531 (1973).
[12] M. O. Dayhoff and R. V. Eck, "Atlas of Protein Sequence and Structure 1967–1968," National Biomedical Research Foundation, Silver Spring, Maryland, 1968.
[13] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *in* "Atlas of Protein Sequence and Structure." (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 345. National Biomedical Research Foundation, Washington, D.C., 1978.
[14] D. T. Jones, W. R. Taylor, and J. M. Thornton, *Comput. Appl. Biosci.* **8,** 275 (1992).
[15] D. T. Jones, W. R. Taylor, and J. M. Thornton, *FEBS Lett.* **339,** 269 (1994).
[16] G. H. Gonnet, M. A. Cohen, and S. A. Benner, *Science* **256,** 1443 (1992).
[17] S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 10915 (1992).

using the formula of Kimura[18] which bases the distance on the fraction of amino acids shared between the sequences, without regard to which amino acids they are. The categories model, as implemented in PROTDIST, can use several different genetic codes (the universal code and several kinds of mitochondrial codes). Three categorizations of the amino acids are used, one the categories given by George *et al.*,[19] one from a categorization in a "baby biochemistry" text, and one the opinion of a colleague. Interestingly, all three of these turn out to be subdivisions of one linear order of amino acids. We have found that a value of $p = 0.45$ brings the ratio of between- to within-category change in the category model of George *et al.*[19] close to that in the Dayhoff model. In the next release (4.0) of PHYLIP, we hope to expand the range of models by including the model of Jones *et al.*[14] and allowing for a gamma distribution of evolutionary rates among sites, in the manner of Jin and Nei[20] and Nei *et al.*[21]

Given the evolutionary model, we use maximum likelihood estimation to compute the distances. In effect we are specifying a two-species tree, with but one branch, between the pair of species, and estimating that branch length by maximum likelihood. If we observe n_{ij} changes between amino acids i and j, and if the model we are using has equilibrium frequency f_i for amino acid i and transition probability $P_{ij}(t)$ over time t, the expected fraction of sites which will have amino acid i in one species and j in the other is $f_i P_{ij}(t)$. The PAM001 matrix gives the conditional probabilities P_{ij}, but they are not reversible. To make a reversible model that is as close as possible to PAM001, we have used instead

$$Q_{ij} = (f_i P_{ij} + f_j P_{ji})/2 \tag{3}$$

This gives us symmetric joint probabilities of observing i and j in two closely related sequences. Suppose that the $\mathbf{M}$ are transition probabilities that would lead to the joint probabilities $\mathbf{Q}$, and that π is the vector of equilibrium frequencies which is implied by $\mathbf{M}$. We start out knowing $\mathbf{Q}$ but not $\mathbf{M}$ or π. It is not hard to show that the eigenvalues of $\pi'\mathbf{M}$ are the same as the eigenvalues of $\mathbf{Q}$, and the eigenvalues of $\mathbf{M}$ can also be directly derived from those of $\mathbf{Q}$. The eigenvalues and eigenvectors of $\mathbf{M}$ are computed in this way (they are precomputed in the PAM001 case and computed by the program in the categories cases).

[18] M. Kimura, "The Neutral Theory of Molecular Evolution." Cambridge Univ. Press, Cambridge, 1983.
[19] D. G. George, W. C. Barker, and L. T. Hunt, this series, Vol. 183, p. 333.
[20] L. Jin and M. Nei, *Mol. Biol. Evol.* **7,** 82 (1990).
[21] M. Nei, R. Chakraborty, and P. A. Fuerst, *Proc. Natl. Acad. Sci. U.S.A.* **73,** 4164 (1976).

From the eigenvalues and eigenvectors of **M** we can readily compute the transition probabilities $M_{ij}(t)$, and their derivatives with respect to t. The likelihood which we must maximize is

$$L = \prod_i \prod_j [\pi_i M_{ij}(t)]^{n_{ij}} \tag{4}$$

The log-likelihood is maximized over values of t by Newton–Raphson iteration.

The resulting distance computation is not fast, but it seems adequate. However, it makes one assumption that is quite severe. All amino acid positions are assumed to change at the same rate. This is unrealistic. To some extent we can compensate for this by correcting the distances by using the approach of Jin and Nei.[20] However there is information that is being lost by doing this. We would like to be able to use the variation in an amino acid position in one part of the data set to infer whether that position allowed change to occur at a high rate, and thus to help us evaluate other parts of the same data set. But no distance matrix method can do this, as they consider only pairs of sequences.

Likelihood Methods

Neyman[3] and Kashyap and Subas[22] developed maximum likelihood methods for inferring phylogenies from protein data. They used the highly oversimplified Jukes–Cantor[10] model of symmetric change among amino acids, and they could not handle more than three or four sequences in the tree in a reasonably exact way. I have shown[23] how to make the likelihood computations practical for larger numbers of species. Likelihood methods for proteins have not been developed further until more recently, because of the computational burden. Where nucleotide sequence likelihood methods use a 4×4 transition probability matrix, in protein models these must be either 20×20 or 64×64, thus requiring either 25 or 256 times as much computation. With increased speed of desktop and laboratory computers, developing a reasonable likelihood method for protein sequences has become more of a priority.

Adachi and Hasegawa[24] and Adachi *et al.*[25] have developed such a method, using the Dayhoff PAM matrix[13] as the transition probability

[22] R. L. Kashyap and S. Subas, *J. Theor. Biol.* **47,** 75 (1974).
[23] J. Felsenstein, *J. Mol. Evol.* **17,** 368 (1981).
[24] J. Adachi and M. Hasegawa, *Jpn. J. Genet.* **67,** 187 (1992).
[25] J. Adachi, Y. Cao, and M. Hasegawa, *J. Mol. Evol.* **36,** 270 (1993).

matrix among amino acid states, but without any direct use of the genetic code. Their program, which is similar to existing DNA likelihood programs but has some effort put into requiring fewer evaluations of the likelihood, is available in their MOLPHY package from their ftp site at sunmh.ism.ac.jp.

It is tempting to develop a method that takes the genetic code explicitly into account. In principle one could have 64 states, one for each codon, and regard the amino acids as ambiguous observations (e.g., alanine would be regarded as an observation of "either TCA or TCG or TCC or TCT"). The computational difficulties would be severe. One could also hope to take into account both observed protein sequence and the underlying DNA sequence, which is often known. Hein[26] and Hein and Støvbæk[27,28] have made a start on such models.

A more serious limitation of existing protein maximum likelihood models is that they assume that all positions change at the same expected rate. This assumption has been removed from nucleotide sequence likelihood models, using hidden Markov model techniques.[29–32] Its extension to proteins is straightforward and badly needed, but it does promise to slow down the computer programs severalfold.

Structure, Alignment, and Phylogeny

Beyond any of these complications is the challenge of taking protein structure into account. Researchers on analysis of RNA sequences have found that there is a synergism between inferences of phylogeny, alignment, and structure. It is just beginning to become widely recognized that the same will be true with proteins, the advantages being probably greater. Structure-based hidden Markov models (HMMs) have been used to improve sequence alignment of proteins, although without taking phylogeny into account.[33,34] Three-dimensional protein structures can be used to infer phylogenies.[35] Structural context affects not only amino acid composition,

[26] J. Hein, *J. Theor. Biol.* **167,** 169 (1994).
[27] J. Hein and J. Støvlbæk, *J. Mol. Evol.* **38,** 310 (1994).
[28] J. Hein and J. Støvlbæk, *J. Mol. Evol.* **40,** 181 (1995).
[29] J. Felsenstein and G. A. Churchill, *Mol. Biol. Evol.* **13,** 93 (1996).
[30] Z. Yang, *Mol. Biol. Evol.* **10,** 1396 (1994).
[31] Z. Yang, *J. Mol. Evol.* **39,** 306 (1994).
[32] Z. Yang, *Genetics* **139,** 993 (1995).
[33] P. Baldi, Y. Chauvin, T. Hunkapiller, and M. A. McClure, *Proc. Natl. Acad. Sci. U.S.A.* **91,** 1059 (1994).
[34] A. Krogh, M. Brown, I. S. Mian, K. Sjölander, and D. Haussler, *J. Mol. Biol.* **235,** 1501 (1994).
[35] M. S. Johnson, A. Šali, and T. L. Blundell, this series, Vol. 183, p. 670.

but the substitution process itself.[36] When residues interact, there may result patterns of compensating substitutions. This has begun to be examined for proteins.[37]

In RNAs, phylogenies and inferences of structure are increasingly important to one another. Patterns of compensating substitutions are strong and have led to mathematical models of this substitution process.[38,39] One can imagine a unified process of inference for proteins and protein-coding regions that simultaneously infers phylogeny, alignment, secondary structure, and three-dimensional structure. The computational problems will be severe, but many of the components needed are already being worked on. Having coordinated our inferences of structure and evolutionary history, we will then be free to dream of considering function as well.

Acknowledgments

This work has been supported by grants from the National Science Foundation (DEB-9207558) and the National Institutes of Health (1 R01 GM 51929-01).

[36] J. Overington, D. Donnelly, M. S. Johnson, A. Šali, and T. L. Blundell, *Protein Sci.* **1**, 216 (1992).
[37] W. R. Taylor and K. Hatrick, *Protein Eng.* **7**, 341 (1994).
[38] E. R. M. Tillier, *J. Mol. Evol.* **39**, 409 (1994).
[39] E. R. M. Tillier and R. A. Collins, *Mol. Biol. Evol.* **12**, 7 (1995).

[25] Reconstruction of Gene Trees from Sequence Data

By NARUYA SAITOU

Properties of Gene Tree

Reconstruction of the phylogeny of genes is essential not only for the study of evolution but also for biology in general because replication of nucleotide sequences automatically produces a bifurcating tree of genes. It should be emphasized that the phylogenetic relationship of genes is different from the mutation process. The former always exists, whereas mutations may or may not happen within a certain time period and DNA region. Therefore, even if several nucleotide sequences happen to be identical, there must be a genealogical relationship for those sequences.

However, it is impossible to reconstruct the genealogical relationship without the occurrence of mutational events. In this respect, the extraction of mutations from genes and their products is important for reconstructing phylogenetic trees of genes. The advancement of molecular biotechnology has made it possible to produce nucleotide sequences routinely. We therefore focus on the analysis of nucleotide sequences. However, a substantial part of this chapter also applies to other molecular data.

Formal Characteristics of Trees

A tree is a kind of graph. A graph is composed of node(s) and branch(es). There should be only one path between any two nodes on a tree. In evolutionary studies, a node represents a gene, species, or population, depending on the purpose, and a branch represents the topological relationship between nodes (often including information on lengths that represent mutational changes or evolutionary time). Nodes are divided into external and internal ones. The former are also called operational taxonomic units (OTUs). There are five OTUs (1–5) and four internal nodes (X, Y, Z, and R1) in the tree in Fig. 1a. Branches are also divided into external and internal ones. An external branch connects an external node and an internal node (e.g., branch 1-X of Fig. 1), whereas an internal branch connects two internal nodes (e.g., branch X-Z of Fig. 1).

A tree can be either rooted or unrooted. A rooted tree has a special node called the root which is defined as the position of the common ancestor

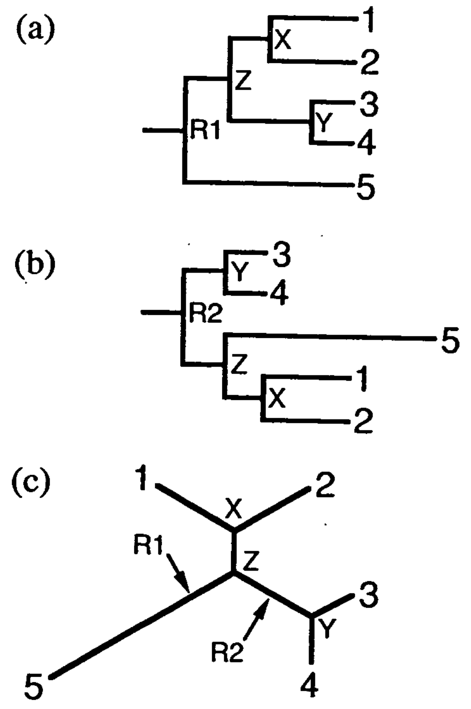


FIG. 1. Examples of rooted trees (a, b) and an unrooted tree (c) for five OTUs.

(see Fig. 1a,b). There will be a unique path from the root to any other node, and the direction of this is, of course, that of time. A phylogenetic tree in an ordinary sense is a rooted tree. Unfortunately, however, many methods for building phylogenetic trees produce unrooted trees, such as the tree of Fig. 1c. An unrooted tree can be converted to a rooted tree if the position of the root is specified. The trees in Fig. 1a,b were produced from the unrooted tree of Fig. 1c.

This relation between rooted and unrooted trees is used for the "outgroup" method of rooting as follows. When we are interested in determining the phylogenetic relationship among n sequences, we will add one (or more) sequence that is known to be an outgroup to the n sequences. The unrooted tree obtained for the $n + 1$ sequences can easily be converted to a rooted tree of n sequences. Sequence 5 corresponds to the outgroup in the tree in Fig. 1c when the root is R1, and the tree of Fig. 1a is then obtained. When the root is R2, sequences 3 and 4 are considered to be the outgroup to sequences 1, 2, and 5, and we obtain the tree of Fig. 1b.

The number of possible tree topologies rapidly increases with an increase in the number of OTUs. The general equation for the possible number of topologies for bifurcating unrooted trees (Tn) for n (≥ 3) OTUs is given by[1]

$$Tn = (2n - 5)!/[2^{n-3}(n - 3)!] \qquad (1)$$

If we apply Eq. (1), there are 221,643,095,476,699,771,875 possible tree topologies for 20 OTUs. It is clear that the search for the true phylogenetic tree of many sequences is a very difficult problem. This is why so many methods have been proposed for building phylogenetic trees.

Gene Trees and Species Trees

Phylogenetic trees of genes and species are called gene trees and species trees, respectively, and there are several important differences between them. One such difference is illustrated in Fig. 2. Because a gene duplication occurred before the speciation of species A and B in Fig. 2a, both species have two homologous genes in their genomes. In this situation, we should distinguish orthology, which is homology of genes reflecting the phylogenetic relationship of species, from paralogy, which is homology of genes caused by gene duplication(s).[2] Thus, genes 1 and 3 (and 2 and 4) are orthologous, whereas genes 1 and 4 (and 2 and 3) are paralogous. If one is not aware of the gene duplication event, the gene tree for 1 and 4

[1] L. L. Cavalli-Sforza and A. W. F. Edwards, *Am. J. Hum. Genet.* **19,** 233 (1967).
[2] W. M. Fitch and E. Margoliash, *Evol. Biol.* **4,** 67 (1970).

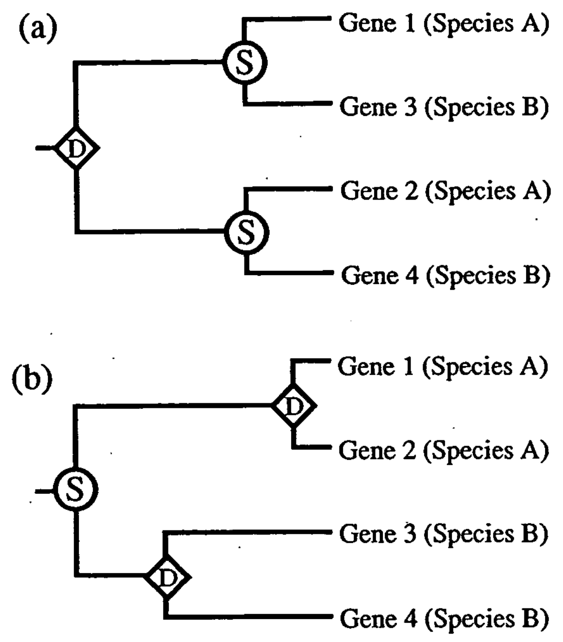


FIG. 2. Two possibilities of a gene tree for four genes sampled from two species. (a) Gene duplication (denoted by D) occurred before speciation (denoted by S). (b) Speciation occurred before two gene duplications.

may be misrepresented as the species tree of A and B, and thus a gross overestimation of the divergence time may occur. It should also be noted that the divergence time between genes 1 and 3 is identical with that between genes 2 and 4, since both times correspond to the same speciation event.

When two homologous gene copies are found in species A and B, another situation is possible, as shown in Fig. 2b. Now two gene duplications occurred after the speciation of species A and B, and two gene copies in the genome of each species are more closely related with each other than the corresponding homologous genes at different species. Because two duplication events occurred independently, the divergence time between genes 1 and 2 is different from that between genes 3 and 4.

Even when orthologous genes are used, a gene tree may be different from the corresponding species tree. This difference comes from the existence of gene genealogy in the ancestral species. A simple example is illustrated in Fig. 3a. A gene sampled from species A has its direct ancestor at the speciation time T_1 generations ago, and so does a gene sampled from species B. Thus, the divergence time between the two genes sampled from the different species always overestimates that of the species. The amount of overestimation corresponds to the coalescence time in the ancestral species, and its expectation is $2N$ for neutrally evolving nuclear genes of

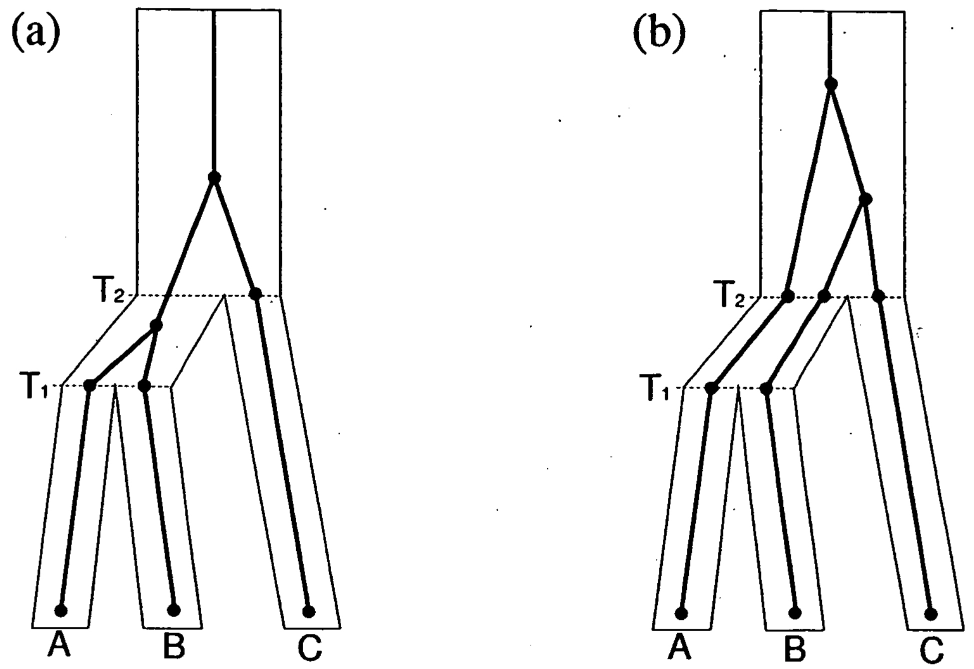


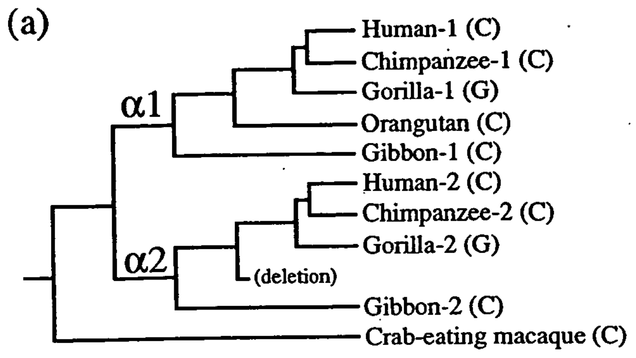
FIG. 3. Difference between a gene genealogy and species tree. (a) Topology of the gene genealogy is the same as that for the species tree. (b) Topology of the gene genealogy is different from that for the species tree. Full circles and thick lines denote the genealogical relationship, whereas thin lines (outlining the gene tree) denote the species tree. A, B, and C denote three genes each sampled from extant species, whereas X and Y denote ancestral genes. T_1 and T_2 denote the two speciation times.

diploid organism, where N is the population size of the ancestral species.[3] Therefore, if the two speciation events (T_1 and T_2) are close enough, the topological relationship of the gene tree may become different from that of the species tree, as shown in Fig. 3b. Although species A and B are more closely related than to C, genes sampled from species B and C happen to be more closely related with each other than to that sampled from species A. The probability (P_{error}) of obtaining an erroneous tree topology is given by[4]

$$P_{\text{error}} = (2/3)\, e^{-T/2N} \tag{2}$$

where $T = T_2 - T_1$ generations. For example, P_{error} is 0.404 when $T = 50{,}000$ and $N = 50{,}000$. Therefore, a species tree estimated from a single gene may not be correct even if the gene tree was correctly estimated. In this case, we should use more than one gene.

[3] F. Tajima, *Genetics* **105,** 437 (1983).
[4] M. Nei, "Molecular Evolutionary Genetics." Columbia Univ. Press, New York, 1987.



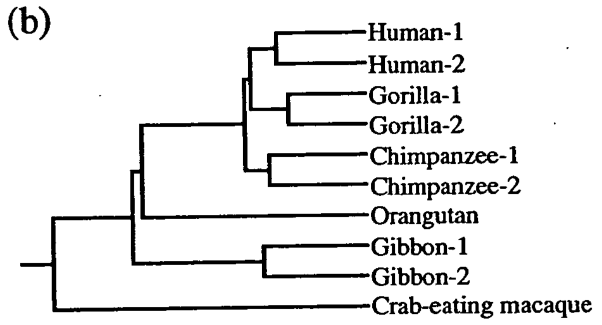


FIG. 4. Alteration of an estimated gene tree caused by gene conversion. (a) The most presumable gene tree for the primate immunoglobulin $\alpha 1$ and $\alpha 2$ genes. C or G in parentheses after species names indicate one nucleotide configuration possibly caused by gene conversion in the gorilla genome. (b) A spurious gene tree (modified from Kawamura *et al.*[5]).

When gene conversion and/or recombination has occurred within the gene region under consideration, a gene tree may be different from the species tree. Kawamura *et al.*[5] examined primate immunoglobulin α genes 1 and 2. Figure 4a shows the plausible gene tree; the gene duplication clearly preceded speciation of hominoids, followed by deletion of the $\alpha 2$ gene from the orangutan genome. However, there are many nucleotide sites that possibly experienced gene conversion. One such example is shown in Fig. 4a; two gorilla genes were both G at a particular nucleotide site, while the remaining genes were C. This suggests either parallel substitution in the gorilla lineage or gene conversion between two gorilla genes. If this kind of nucleotide configuration is contiguous, gene conversion is suspected. The resulting spurious gene tree (Fig. 4b) is distorted from the tree of Fig. 4a because of the strong effect of gene conversion.

Ideally, branch lengths of a phylogenetic tree are proportional to the

[5] S. Kawamura, N. Saitou, and S. Ueda, *J. Biol. Chem.* **267,** 7359 (1992).

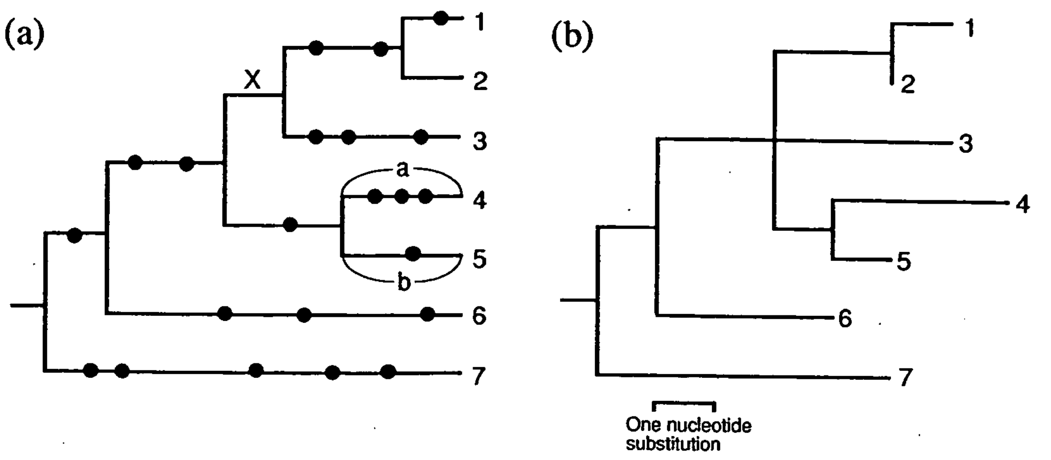


FIG. 5. Examples of the expected gene tree (a) and the corresponding realized gene trees (b). Filled circles on the expected gene tree denote nucleotide substitutions. Because no substitution occurred at branch X of the expected gene tree (a), the corresponding branch does not exist in the realized gene tree (b).

physical time since divergence. Thus the branch a and b of Fig. 5a should be the same length. We call this type of rooted tree the expected tree.[4] Both species and gene trees have their expected trees, but their properties are somewhat different from each other. An expected gene tree directly reflects the history of DNA replications, whereas an expected species tree is a gross simplification of the course of differentiation of populations. Therefore, the speciation time is always unclear.

As mentioned earlier, the genealogical relationship of genes, or expected gene tree, is independent from the mutation process. However, mutation events are essential for the reconstruction of phylogenetic trees. Thus, we can at best estimate a gene tree according to the mutation events realized on its expected gene tree. We call this ideal reconstruction of the gene tree the realized gene tree (Fig. 5b), whereas the reconstructed one from observed data is called the estimated gene tree.[6] Branch lengths of realized and estimated gene trees are proportional to mutational events. These mutational events are not necessarily proportional to physical time. By definition, expected gene trees are strictly bifurcating, while realized and estimated gene trees may be multifurcating. This is because of the possibility of no mutation at a certain branch, such as branch X of Fig. 5a.

A species tree reconstructed from observed data is called an estimated species tree, but there is no realized species tree. It should also be noted that both expected and realized trees are rooted, while estimated trees are often unrooted due to the limitations of available information.

[6] N. Saitou, *in* "Molecular Biology: Current Innovations and Future Trends Part 2" (H. G. Griffin and A. M. Griffin, eds.), p. 115. Horizon Scientific Press, Norfolk, England, 1995.

TABLE I
CLASSIFICATION OF TREE-MAKING METHODS

Method	Stepwise clustering	Exhaustive search
Distance matrix	UPGMA	KITCH
	Distance Wagner	Fitch–Margoliash
	Neighbor joining	Minimum evolution
Character state		Maximum parsimony
		Compatibility
		Maximum likelihood

Methods for Building Phylogenetic Trees of Genes

Classification of Tree-Building Methods

Many methods have been proposed for building a phylogenetic tree from observed data. To clarify the nature of each method, it is useful to classify these methods from various aspects. Tree-building methods can be divided into two types in terms of the type of data they use: distance matrix methods and character-state methods. A distance matrix consists of all the possible pairwise distances, whereas an array of character states is used for the character-state methods. UPGMA (unweighted pairgroup method using arithmatic mean),[7] the Fitch and Margoliash method,[8] the distance Wagner method,[9] the neighbor-joining method,[10] and the minimum evolution methods[1,11,12] are distance matrix methods, whereas the maximum parsimony method,[13] the compatibility method,[14] and the maximum likelihood method[15] are character-state methods (Table I).

Another classification is by the strategy of a method to find the best tree. One way is to examine all or a large number of possible tree topologies and choose the best one according to a certain criterion. We call this the exhaustive search method. The other strategy is to examine a local topological relationship of OTUs and find the best tree. These types of methods are called stepwise clustering methods. Both strategies are used for the distance matrix methods, while the exhaustive search strategy is usually used for character-state methods (Table I).

[7] P. H. P. Sneath and R. Sokal, "Numerical Taxonomy." Freeman, San Francisco, 1977.
[8] W. M. Fitch and E. Margoliash, *Science* **155,** 279 (1967).
[9] J. S. Farris, *Am. Nat.* **106,** 645 (1972).
[10] N. Saitou and M. Nei, *Mol. Biol. Evol.* **4,** 406 (1987).
[11] N. Saitou and T. Imanishi, *Mol. Biol. Evol.* **6,** 514 (1989).
[12] A. Rzhetsky and M. Nei, *Mol. Biol. Evol.* **9,** 945 (1992).
[13] W. M. Fitch, *Am. Nat.* **111,** 223 (1977).
[14] W. J. Le Quesne, *Syst. Zool.* **18,** 201 (1969).
[15] J. Felsenstein, *J. Mol. Evol.* **17,** 368 (1981).

TABLE II
ESTIMATED NUMBER OF NUCLEOTIDE SUBSTITUTIONS PER SITE BETWEEN EVERY PAIR OF 10 SEQUENCES[a]

	1	2	3	4	5	6	7	8	9
2	0.0516								
3	0.0550	0.0031							
4	0.0483	0.0221	0.0253						
5	0.0582	0.0651	0.0685	0.0549					
6	0.0094	0.0416	0.0450	0.0384	0.0549				
7	0.0125	0.0584	0.0619	0.0551	0.0651	0.0157			
8	0.0284	0.0687	0.0722	0.0654	0.0754	0.0317	0.0285		
9	0.0925	0.1221	0.1259	0.1185	0.1370	0.0820	0.0786	0.0927	
10	0.1921	0.2183	0.2228	0.2054	0.2309	0.1798	0.1795	0.1833	0.1860

[a] Gaps were eliminated from the comparison, and a total of 323 nucleotide sites were compared. Kimura's two-parameter method was used [M. Kimura, *J. Mol. Evol.* **16,** 111 (1980)]. Sequence identifications: 1, *Mus mus domesticus* functional gene; 2, *M. mus domesticus* pseudogene; 3, *M. mus castaneus* pseudogene; 4, *M. spicilegus* pseudogene; 5, *M. leggada* pseudogene; 6, *M. mus domesticus* cDNA; 7, *M. leggada* functional gene; 8, *M. platythrix* functional gene; 9, *Rattus norvegicus* cDNA; 10, *Homo sapiens* cDNA.

In distance matrix methods, a phylogenetic tree is constructed by considering the relationship among the distance values D_{ij} (distance between OTUs i and j). An example of a distance matrix is presented in Table II. The distances were computed from the nucleotide sequences for ten p53 functional genes and pseudogenes.[16] These sequence data will be used consistently in this chapter for worked-out examples. There are many methods for estimating evolutionary distances from nucleotide sequences.

Because there are already many reviews on tree-building methods,[4,6,17,18] we describe only the following six methods; UPGMA, the neighbor-joining method, the minimum evolution method, the maximum parsimony method, the maximum likelihood method, and network methods.

Methods Assuming Molecular Clock

When constancy of the evolutionary rate, or a molecular clock, is assumed, we can reconstruct rooted trees. This is because sequences should

[16] H. Ohtsuka, M. Oyanagi, Y. Mafune, N. Miyashita, T. Shiroishi, K. Moriwaki, R. Kominami, and N. Saitou, *Mol. Phylogenet. Evol.* in press.

[17] N. Saitou, *in* "Handbook of Statistics, Volume 8: Statistical Methods for Biological and Medical Sciences" (C. R. Rao and R. Chakraburty, eds.), p. 317. Elsevier, Amsterdam, 1990.

[18] D. L. Swofford and G. J. Olsen, *in* "Molecular Systematics" (D. M. Hillis and C. Moritz, eds.), p. 411. Sinauer Associates, Sunderland, Massachusetts, 1990.

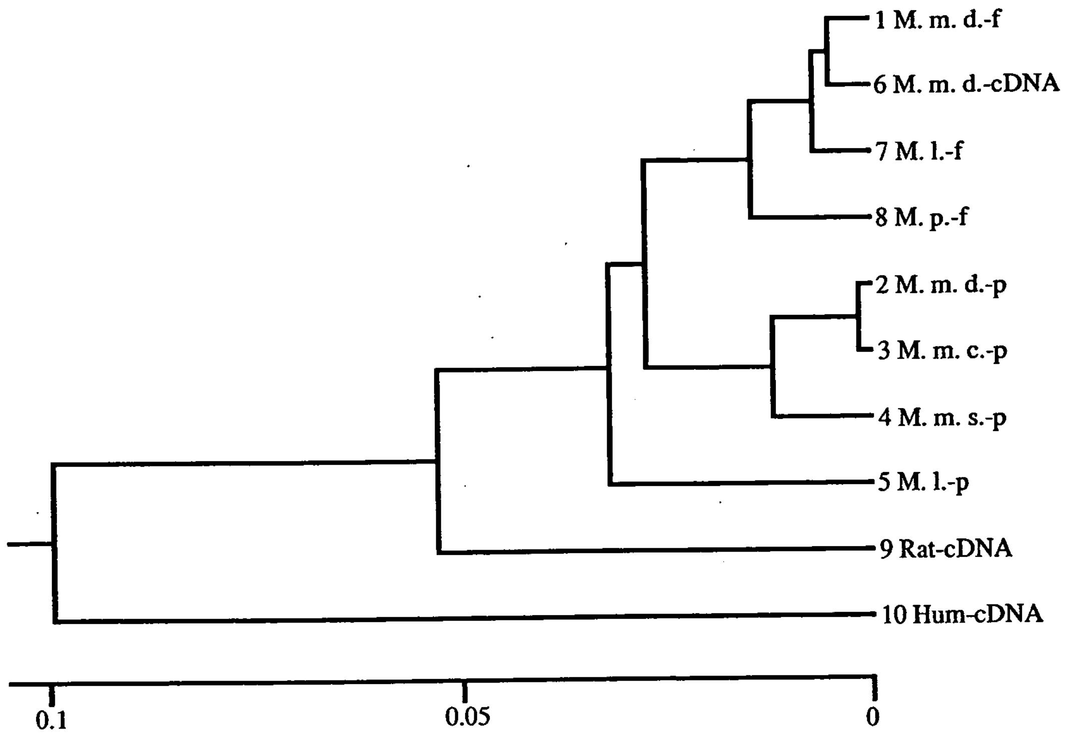


FIG. 6. UPGMA tree for the distance matrix in Table II. Sequence identifiers correspond to those of Table II.

be clustered in the order of their mutational difference if the amount of mutational changes is strictly proportional to evolutionary time. There are many ways to obtain such rooted trees from a distance matrix.[7] In this section, UPGMA and KITCH are discussed.

Let us briefly explain the UPGMA algorithm using the distance matrix shown in Table II. We first choose the smallest distance, D_{23} (=0.0031). Then OTUs 2 and 3 are combined and the distances between the combined OTU [2–3] and the remaining eight OTUs are computed by taking arithmetic means. At the next step, again the smallest distance (D_{16} = 0.0094) is chosen from the distance matrix. Then the OTUs 1 and 6 are combined into OTU [1–6]. This process is continued until all the OTUs are finally clustered into a single one. The resultant rooted tree is shown in Fig. 6. Although *Mus* functional genes (sequences 1, 6, 7, and 8) formed a monophyletic cluster, the corresponding *Mus* pseudogenes (sequences 2–5) did not form a monophyletic one.

Because of the long history of UPGMA (originally proposed by Sokal and Michener[19]), there are many computer programs available for UPGMA, and these are not specified. It should be noted that there are several

[19] R. Sokal and C. D. Michener, *Univ. Kansas Sci. Bull.* **28,** 1409 (1958).

synonyms for UPGMA, such as the simple linkage method, the clustering method, and the nearest neighbor method.

KITCH is a computer program in the PHYLIP package[20] and is related to the Fitch and Margoliash method,[8] but constancy of the evolutionary rate is assumed. Because of this restriction, the result of KITCH is usually quite close to that of UPGMA. In fact, when KITCH was applied to the distance matrix of Table II, a result (not shown) identical to that of UPGMA was obtained. It seems that there is no use for this exhaustive search program if one already has the result using a UPGMA program.

A simulation study[10] has shown that UPGMA is not efficient in reconstructing the true topological relationship when the constancy of evolutionary rate is not assumed. Therefore, it is not advisable to use methods assuming a molecular clock for estimating realized trees. However, those are still useful for estimating expected trees, where all the branch lengths are proportional to physical time.

When we have only an unrooted tree with no outgroup, there is a way of rooting it if we assume a rough constancy of the evolutionary rate. Given the unrooted tree topology, we successively cluster OTU pairs starting from the smallest distance similar to UPGMA.[21] If we apply this algorithm to the unrooted tree of Fig. 1c, we will obtain the tree of Fig. 1a.

Neighbor-Joining Method

A pair of OTUs are called neighbors when these are connected through a single internal node in an unrooted bifurcating tree. For example, OTUs 1 and 2 of Fig. 1c are a pair of neighbors. If we combine these OTUs, this combined OTU [1–2] and OTU 5 become a new pair of neighbors. It is thus possible to define the topology of a tree by successively joining pairs of neighbors and producing new pairs of neighbors. In general, $n - 3$ pairs of neighbors are necessary to define the topology of an unrooted tree with n OTUs.

The neighbor-joining method[10] produces a unique final unrooted tree by sequentially finding pairs of neighbors by examining a distance matrix. Thus the neighbor-joining method is a distance matrix method as well as a stepwise clustering method. The principle of minimum evolution is used in the neighbor-joining method, and it has been proved that the expected value of the sum of branch lengths is smallest for the tree with the true branching pattern.[22] Because of the simple algorithm, more than 100 OTUs

[20] J. Felsenstein, "PHYLIP: Phylogeny Inference Package, Version 3.5c." Univ. of Washington, Seattle, 1993.

[21] N. Ishida, T. Oyunsuren, S. Mashima, H. Mukoyama, and N. Saitou, *J. Mol. Evol.* **41**, 180 (1995).

[22] A. Rzhetsky and M. Nei, *Mol. Biol. Evol.* **10**, 1073 (1993).

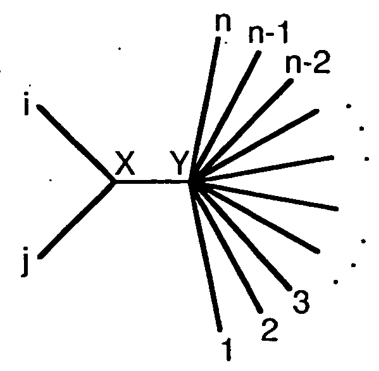


FIG. 7. Tree of N OTUs in which OTUs i and j are neighbors.

can be handled within a relatively short computer time by using the neighbor-joining method. For example, Horai *et al.*[23] produced a neighbor-joining tree for 193 human mitochondrial DNA sequences.

The following explanation of the neighbor-joining algorithm is based on Saitou.[6] We start from a starlike tree, which is produced under the assumption of no clustering among all the n OTUs compared. Under this tree, the sum (S_0) of n branch lengths can be shown to be

$$S_0 = Q/(n-1) \qquad (3)$$

where

$$Q = \Sigma_{i<j} D_{ij} \qquad (4)$$

In practice, some pairs of OTUs are more closely related to one another than other pairs are. Among all the possible pairs of OTUs $[n(n-1)]/2$ pairs for n OTUs], we choose the one that gives the smallest sum of branch lengths. Let us consider the tree of Fig. 7, where OTUs i and j are assumed to be neighbors. The sum of branch lengths is defined by

$$S_{ij} = (B_{iX} + B_{jX}) + B_{XY} + \Sigma_{k\neq i,j} B_{kY} \qquad (5)$$

where $B_{\alpha\beta}$ is branch length between nodes α and β. There are the following relationships between distances and branch lengths:

$$D_{ij} = B_{iX} + B_{jX} \qquad (6a)$$
$$D_{ik} = B_{iX} + B_{XY} + B_{kY} \qquad (k \neq i,j) \qquad (6b)$$
$$D_{jk} = B_{jX} + B_{XY} + B_{kY} \qquad (k \neq i,j) \qquad (6c)$$
$$D_{kl} = B_{iY} + B_{jY} \qquad (k,l \neq i,j) \qquad (6d)$$

[23] S. Horai, R. Kondo, Y. Nakagawa-Hattori, S. Hayashi, S. Sonoda, and K. Tajima, *Mol. Biol. Evol.* **10,** 23 (1993).

With the tree shown in Fig. 7, it can be shown by applying the above relationship that

$$B_{XY} = [Q - (n-1)D_{ij} - (n-1)\,\Sigma_{k,l\neq i,j}\, D_{kl}/(n-3)]/2(n-2) \qquad (7)$$

If we neglect OTUs i and j in Fig. 7, the remaining $n - 2$ OTUs form a starlike tree, as is clear from Eq. (6d). Thus we apply Eq. (3) and obtain

$$\Sigma_{k\neq i,j}\, B_{kY} = \Sigma_{k,l\neq i,j}\, D_{kl}/(n-3) \qquad (8)$$

We also note that

$$\Sigma_{k,l\neq i,j}\, D_{kl} = Q - (R_i + R_j - D_{ij}) \qquad (9)$$

where $R_i = \Sigma_j D_{ij}$ and $R_j = \Sigma_i D_{ij}$. Putting Eqs. (6a), (7), and (8) into Eq. (5) with consideration of Eq. (9), we obtain

$$S_{ij} = D_{ij}/2 + [2Q - R_i - R_j]/2(n-2) \qquad (10)$$

Equation (10) was first proposed by Studier and Keppler.[24]

This S_{ij} value is computed for all $n(n - 1)/2$ pairs of OTUs, and the pair that has the smallest S_{ij} value is chosen as neighbors. This pair of OTUs is then regarded as a single OTU, and the new distances between the combined OTU and the remaining ones are computed by averaging. This procedure is continued until all pairs of neighbors are found.

If OTUs i and j are chosen as neighbors as shown in Fig. 7, the branch lengths are estimated using the Fitch and Margoliash procedure[8] as

$$B_{iX} = D_{ij}/2 + (R_i - R_j)/2(n-2) \qquad (11a)$$

and

$$B_{jX} = D_{ij} - B_{iX} \qquad (11b)$$

Therefore, all the branch lengths as well as the tree topology will be determined after $n - 2$ steps for n OTUs.

Table III shows the output of the computer program NJNUC, and Fig. 8 shows the neighbor-joining tree. Human p53 cDNA sequence (OTU 10) was assumed to be the outgroup. Branch lengths are estimated numbers of nucleotide substitutions that occurred in this p53 sequence, and all of them are integer values. To obtain those numbers, estimated numbers of nucleotide substitutions per site (numbers in parentheses in Table III) were multiplied with the number of compared nucleotide sites, then the resulting values were rounded. If a branch length turned out to be zero,

[24] J. A. Studier and K. J. Keppler, *Mol. Biol. Evol.* **5**, 729 (1988).

TABLE III
OUTPUT OF PROGRAM NJNUC FOR p53 SEQUENCE DATA[a]

Node 11	OTU 9 = 14.632 (4.530E-02)	OTU 10 = 45.440 (1.407E-01)
Node 12	OTU 2 = −0.065 (−2.011E-04)	OTU 3 = 1.069 (3.311E-03)
Node 13	Node 12 = 4.463 (1.382E-02)	OTU 4 = 2.693 (8.339E-03)
Node 14	Node 13 = 4.281 (1.325E-02)	OTU 5 = 11.547 (3.575E-02)
Node 15	OTU 8 = 5.427 (1.680E-02)	Node 11 = 9.108 (2.820E-02)
Node 16	OTU 7 = 1.913 (5.923E-03)	Node 15 = 1.235 (3.822E-03)
Node 17	Node 14 = 5.102 (1.580E-02)	OTU 6 = 0.532 (1.646E-03)
Node 18	(Last node)	
OTU 1	1.675 (5.186E-03)	
Node 17	0.907 (2.808E-03)	
Node 16	1.410 (4.364E-03)	

[a] The distance matrix of Table II was used. Numbers after the OTU or node designation are branch lengths in terms of nucleotide substitutions that occurred at the compared sequence region between that node/OTU and the node written at the top of each row. Numbers in parentheses are branch lengths in terms of nucleotide substitutions per site.

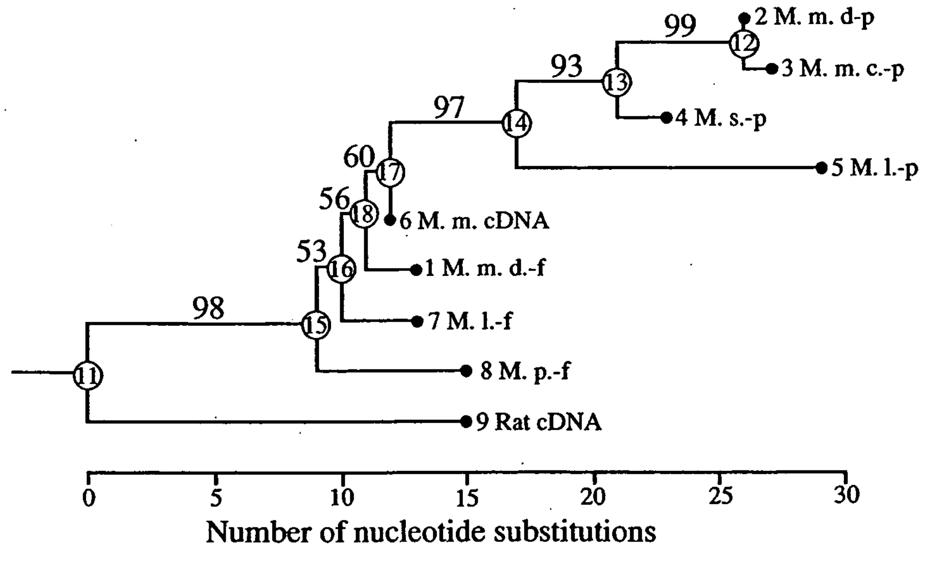


FIG. 8. Neighbor-joining tree constructed from the distance matrix of Table III. This tree was drawn on the basis of the output shown in Table IV. Branch lengths are proportional to the number of nucleotide substitutions per branch. Sequence identifiers correspond to those of Table II, and numbers in circles are internal node identifications. Numbers above internal branches are bootstrap probabilities (%). Human cDNA sequence was assumed to be the outgroup.

such as the branch 2–12, that branch was truncated. A tree obtained by applying this procedure, first proposed by Nerurker *et al.*,[25] is an estimation of the realized tree. The topology of the neighbor-joining tree (Fig. 8) is somewhat different from that of the UPGMA tree (Fig. 6). Now the *Mus* pseudogenes (sequences 2–5) form a monophyletic cluster, while *Mus* functional counterparts (sequences 1, 6, 7, and 8) do not form a monophyletic cluster.

Numbers above internal nodes in the tree of Fig. 8 are bootstrap probabilities (percentages) based on 1000 replications (program NJBOOT2 was used for obtaining the bootstrap probabilities). For example, all the four pseudogene sequences are clustered with a high bootstrap probability (97%) at the internal node C. The bootstrap method was proposed for estimating variances from unknown probability distributions[26] and was introduced into phylogenetic study.[27] Character-state data are necessary to use the bootstrap method, but trees built using any distance matrix method can be tested using this technique. We first randomly resample n nucleotide sites from the given sequence data of n nucleotides with replacement. This resampling is replicated at least 1000 times. For example, one replication may have n nucleotide sites with the positions $1, 2, 2, 4, 5, \ldots, n-2, n$, and n. Resampling is usually done by generating pseudorandom numbers. Each replicated sequence data set is then used as the input data to build phylogenetic trees. A bootstrap probability of a certain internal branch is simply the number of trees that realize this branch divided by the total number of replications. These probabilities are often summarized on the phylogenetic tree estimated by using the original sequence data. The bootstrap method is currently widely used, but its influence on phylogenetic inference is not thoroughly known; theoretical studies are still going on.

A series of programs (NJ, NJNUC, etc.) are included in the TreeTree package developed by the author. This method is also available in packages PHYLIP,[20] MEGA,[28] CLUSTAL W,[29] and MOLPHY.[30] Programs NJBOOT2 and TREEVIEW run on MS-DOS developed by K. Tamura (E-mail: Koichiro-Tamura@c.metro-u.ac.jp) are also available.

[25] V. R. Nerurkar, K.-J. Song, N. Saitou, R. R. Mallan, and R. Yanagihara, *Virology* **196**, 506 (1993).

[26] B. Efron, *Ann. Stat.* **7**, 1 (1979).

[27] J. Felsenstein, *Evolution* **39**, 783 (1985).

[28] S. Kumar, K. Tamura, and M. Nei, "MEGA: Molecular Evolutionary Genetics Analysis, Version 1.0." The Pennsylvania State Univ., University Park, 1993.

[29] J. D. Thompson, D. G. Higgins, and T. J. Gibson, *Nucleic Acids Res.* **22**, 4673 (1994).

[30] J. Adachi and M. Hasegawa, "MOLPHY: Programs for Molecular Phylogenetics, Version 2.2." Institute of Statistical Mathematics, Tokyo, 1994.

Minimum Evolution Methods

The concept of minimum evolution was used in the neighbor-joining method, and this concept was first used by Cavalli-Sforza and Edwards.[1] Saitou and Imanishi[11] proposed a simple method applying the principle of minimum evolution. In this method, branch lengths of a given tree are estimated by applying the procedure of Fitch and Margoliash,[8] and the tree with the smallest sum of branch lengths is chosen as the best tree. Rzhetsky and Nei[12] proposed a minimum evolution method in which branch lengths with their standard errors are computed by applying the least squares method. A neighbor-joining tree is first constructed as the candidate tree, and the related trees with only small topological differences are then searched. A simplified algorithm for computing least squares estimates of branch lengths has been proposed to reduce the computation time.[22]

Table IV shows an output of a minimum evolution program ME_TREE. There is one topological difference between the neighbor-joining tree (see Fig. 8) and this minimum evolution tree, regarding the clustering of sequences 1 and 7. The sum of branch lengths are 0.346188 and 0.346526 for the minimum evolution and neighbor-joining trees, respectively. Standard errors of branch lengths are small when the bootstrap values of the corresponding branches (see Fig. 8) are high. For example, the internal branch

TABLE IV
OUTPUT OF PROGRAM ME_TREE FOR p53 SEQUENCE DATA[a]

Branch	Branch length ± SE	Significance level (%)
1 and 16	0.005795 ± 0.003568	89.48
2 and 12	−0.000243 ± 0.000247	67.30
3 and 12	0.003349 ± 0.003354	67.78
4 and 13	0.008107 ± 0.005287	87.14
5 and 14	0.036693 ± 0.011213	99.90
6 and 17	0.001429 ± 0.002244	47.14
7 and 16	0.006695 ± 0.003946	90.90
8 and 15	0.017395 ± 0.006561	99.20
9 and 11	0.045297 ± 0.012698	99.96
10 and 11	0.140679 ± 0.023459	99.96
11 and 15	0.027608 ± 0.011227	98.58
12 and 13	0.014052 ± 0.006792	96.06
13 and 14	0.013044 ± 0.007296	92.50
14 and 17	0.014755 ± 0.006999	96.42
15 and 18	0.005024 ± 0.004819	70.16
16 and 18	0.002454 ± 0.004777	39.00
17 and 18	0.004056 ± 0.002930	83.24

[a] The distance matrix of Table II was used.

TABLE V
CLASSIFICATION OF NUCLEOTIDE CONFIGURATIONS OF p53 SEQUENCE DATA OF 323 NUCLEOTIDES FOR MAXIMUM PARSIMONY METHOD

Category	Observed number	Minimum number of changes
Noninformative configuration		
Invariant	236	0
Variant with 2 nucleotides	50	50
Variant with 3 nucleotides	9	18
Variant with 4 nucleotides	0	0
Informative configuration		
Variant with 2 nucleotides	26	26
Variant with 3 nucleotides	2	4
Variant with 4 nucleotides	0	0
Total	323	98

(0.014755 ± 0.006999) connecting nodes 14 and 18 is significantly larger than zero (significance level of 96.42%), and the corresponding bootstrap probability for the neighbor-joining tree is 97%.

There is a computer program (ME_TREE) run on MS-DOS.[31] Another program run on SUN workstations has been developed by Igor Belyi [WWW (World Wide Web) home page is http://www.cse.psu.edu/~belyi].

Maximum Parsimony Methods

The principle of maximum parsimony was first used for morphological data,[32] but it was independently proposed also for molecular data.[33] There are several kinds of maximum parsimony methods based on various assumptions, but the one that produces unrooted trees as in the case of the neighbor-joining method is mainly used for nucleotide sequence data.[13] The maximum parsimony principle is the minimization of the character-state changes (tree length) on the given tree topology, and is related to the principle used in minimum evolution methods. However, the performance of the two methods in choosing the best topology can be quite different.

Let us consider the example sequence data. We first classify the 323 nucleotide sites into different configurations (Table V). A nucleotide configuration is a distribution pattern of nucleotides for a given number of

[31] A. Rzhetsky and M. Nei, *Comput. Appl. Biosci.* **10,** 409 (1994).
[32] J. H. Camin and R. Sokal, *Evolution* **19,** 311 (1965).
[33] R. V. Eck and M. O. Dayhoff, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff ed.). National Biomedical Research Foundation, Silver Spring, Maryland, 1966.

sequences. The possible number (C_n) of configurations for n sequences is given by[34]

$$C_n = (4^{n-1} + 3 \times 2^{n-1} + 2)/6 \tag{12}$$

For example, there are 51 possible nucleotide configurations for five sequences. It should be noted that the number of possible configuration increases if we distinguish transitional differences from transversional ones.

Those configurations are first divided into noninformative and informative ones (Table V). Configurations that do not contribute to the selection of tree topology are called noninformative for the maximum parsimony method. All the sequences have the same nucleotide at the invariant configuration. There were 236 sites that fell into this category. We do not need any nucleotide substitution for this configuration under the maximum parsimony principle. One and two substitutions are necessary for any topology for variant with two and three nucleotides of the noninformative configuration, respectively. An informative nucleotide configuration should have more than one kind of nucleotide, and at least two of these should be observed in more than one of the sequences.[13] Only 28 of 323 sites had informative configurations. In total, we need at least 98 nucleotide substitutions for this data set.

Because there already exist several descriptions of the maximum parsimony method,[4,6,17,18] we will skip the explanation of the method in this chapter and show only the worked-out example. The result of the maximum parsimony analysis using PAUP is presented in Table VI. Nine equally parsimonious trees that require 112 substitutions were found by using branch-and-bound as well as heuristic options. However, two of them turned out to be identical with each other if we truncate an internal branch with zero length. Thus, the real number of equally parsimonious trees was eight (trees 1–8 of Table VI). Tree 6 had the same topology with the neighbor-joining tree (Fig. 8), and tree 4 was the maximum likelihood tree. Trees 9–12 required 113 substitutions and thus are subparsimonious. Biologically, however, tree 10 or 11 seems to be more reasonable.[16] It is also interesting to note that the minimum evolution tree (tree 12 of Table VI) was not a maximum parsimonious tree.

The principle of maximum parsimony attracted many because of its simplicity and logical clarity. However, there are some problems with this method when molecular data are used. Felsenstein[35] showed analytically that the maximum parsimony method may be positively misleading when the rate of evolution is grossly different among lineages of four sequences.

[34] N. Saitou and M. Nei, *J. Mol. Evol.* **24,** 189 (1986).
[35] J. Felsenstein, *Syst. Zool.* **27,** 401 (1978).

TABLE VI
MAXIMUM PARSIMONY AND MAXIMUM LIKELIHOOD ANALYSES

Tree ID[a]	Tree topology[b]	RNM[c]	Differences of log L[d]
1	(((((((2,3),4),5),8),6),1),7,(9,10))	112	−4.07
2	((((7,8),1),6),(((2,3),4),5),(9,10))	112	−0.29
3	(1,6,(((((2,3),4),5),8),((9,10),7)))	112	−4.09
4	(((((((2,3),4),5),6),1),8),7,(9,10))	112	Best
5	((((((2,3),4),5),(6,1)),8),7,(9,10))	112	−5.39
6	(((((((2,3),4),5),6),1),7),8,(9,10))	112	−2.55
7	(((((2,3),4),5),(6,(1,7))),8,(9,10))	112	−4.31
8	(((((2,3),4),5),6),((1,7),8),(9,10))	112	−6.10
9	(((1,6),7),((((2,3),4),5),8),(9,10))	113	−9.02
10	((((1,6),7),(((2,3),4),5)),8,(9,10))	113	−8.80
11	((((1,6),7),8),(((2,3),4),5),(9,10))	113	−8.99
12	((((((2,3),4),5),6),(1,7)),8,(9,10))	113	−7.37

[a] Trees 6 and 12 are the neighbor-joining tree (Fig. 8) and the minimum evolution tree (Table IV), respectively.
[b] Sequence identifications are the same as those of Table II.
[c] Required number of mutations when the maximum parsimony method was applied.
[d] Differences of log likelihood values from that of the best tree (tree 4; its log likelihood was −1016.75).

When the expected number of required substitutions for the true tree is larger than that for a wrong one, the maximum parsimony method will give more and more wrong answers as the number of compared nucleotides is increased (problem of efficiency). The same problem was found even when constancy of the evolutionary rate is assumed.[36,37] Saitou[38] showed that the gross underestimation of the branch lengths occurred when the divergence (number of nucleotide substitutions per site) among sequences was larger than 0.2. Therefore, we should be careful when using the maximum parsimony method.

After the tree topology is determined, however, the principle of maximum parsimony can be useful for estimating the location of mutational events. For example, Gojobori *et al.*[39] estimated the direction of nucleotide substitutions, and Saitou and Ueda[40] mapped the insertions and deletions on the assumed phylogenetic tree of primates. Jermann *et al.*[41] have recon-

[36] A. Zharkikh and W.-H. Li, *Syst. Biol.* **42,** 113 (1993).
[37] N. Takezaki and M. Nei, *J. Mol. Evol.* **39,** 210 (1994).
[38] N. Saitou, *Syst. Zool.* **38,** 1 (1989).
[39] T. Gojobori, W.-H. Li, and D. Graur, *J. Mol. Evol.* **18,** 360 (1982).
[40] N. Saitou and S. Ueda, *Mol. Biol. Evol.* **11,** 504 (1994).
[41] T. M. Jermann, J. G. Opitz, J. Stachkouse, and S. A. Benner, *Nature (London)* **374,** 57 (1995).

structed ancestral ribonuclease proteins from the estimated tree for the artiodactyls. It should be noted that the maximum parsimony principle can be applied to any tree irrespective of the methods used for constructing it.

PAUP 3.1.1 (a commercial product distributed from Illinois Natural History Servey) is run on a Macintosh with many user-friendly options. Maximum parsimony analysis is possible also for PHYLIP[20] and MEGA.[28] MacClade[42] has various useful features for molecular data, although it does not search the topology space.

Maximum Likelihood Methods

The maximum likelihood method is often used for parameter estimation in statistics, and it was first applied to building phylogenetic trees for allele frequency data.[1] Later, various maximum likelihood methods and computer programs were developed for sequence data.

The core algorithm of the maximum likelihood method is as follows. We first define the probability $P_{\alpha\beta} \equiv Pr(N_\alpha, N_\beta, B_{\alpha\beta})$ for observing nucleotides N_α and N_β at a particular nucleotide site at nodes α and β, respectively, when branch length is $B_{\alpha\beta}$. It is necessary to define the nucleotide transition matrix to compute $P_{\alpha\beta}$. Because each nucleotide site is assumed to evolve independently, the likelihood values for all the nucleotide sites are multiplied to obtain the overall likelihood. As is usually done in maximum likelihood techniques, the logarithm of the likelihood (log L) is computed by changing branch lengths, and the maximum likelihood solution is determined for this tree topology. This maximum likelihood solution is ideally obtained for all the possible topologies, and the one that shows the highest value is chosen. Interested readers may refer to more detailed descriptions of this method.[4,6,15,43]

Table VI shows the result of the DNAML computation (user tree option was used). Tree 4, one of 8 equally parsimonious trees, was found to have the highest likelihood among the 12 trees compared. Likelihood values for subparsimonious trees (with 113 required substitutions) are somewhat lower than those for equally parsimonious trees.

Because the maximum likelihood method requires massive computer time, there are several searching methods other than the exhaustive search. The default method of DNAML[15,20] is the sequential addition of sequences. Saitou[44] proposed a stepwise clustering of sequences for the maximum likelihood method, and this searching method is the same as that of the

[42] W. P. Maddison and D. R. Maddison, "MacClade Version 3." Sinauer Associates, Sunderland, Massachusetts, 1992.
[43] N. Saitou, this series, Vol. 183, p. 584.
[44] N. Saitou, *J. Mol. Evol.* **27,** 261 (1988).

neighbor-joining method. NucML of MOLPHY[30] has several options for topology searches, and one of them (star decomposition) is similar to that of the Saitou[44] method.

DNAML and DNAMLK (molecular clock is assumed) are included in PHYLIP.[20] There is also a modified version of DNAML called fast-DNAML.[45] NucML for nucleotide sequences and ProtML for amino acid sequences are included in MOLPHY.[30]

Methods Producing Networks, Not Trees

The evolutionary history of a gene should be presented as a tree. When we analyze real sequence data, however, this tree structure may not be clearly observed. Bandelt and Dress[46] proposed the split decomposition method for distance matrix data. Unlike most tree-building methods, it usually produces a network, not a tree. A relaxed condition is used for estimating splitting patterns among OTUs, and both the signal (suggesting the tree structure) and the noise (suggesting patterns inconsistent with the tree structure) can be presented simultaneously. The resultant network is shown in Fig. 9. Four parallelograms suggest the existence of some parallel nucleotide changes, though the overall structure is quite close to an unrooted tree.

This network construction can also be applied to sequence data directly. When two nucleotide positions show an incongruent partition pattern, a discordancy diagram[13] appears. Bandelt[47] has extended this idea and proposed the phylogenetic network method. A network structure is useful for delineating anomaly in the history of gene trees. For example, when two regions of a gene experienced recombination(s), we may obtain a network, not a tree, if we analyze the sequence data by combining the two regions.

Regarding program availability, Daniel Huson (huson@mathematik.uni-bielefeld.de) and Rainer Wetzel have developed a shareware called SplitsTree run on Macintosh. A program for the phylogenetic network method is under development by H.-J. Bandelt.

Freely Distributed Computer Packages

PHYLIP[20] contains many programs in the form of both source code and executable files. Various kinds of maximum likelihood methods, maximum parsimony methods, and distance matrix methods can be used. It can be retrieved from evolution.genetics.washington.edu (128.95.12.41) or from

[45] G. J. Olsen, H. Matsuda, R. Hagstrom, and R. Overbeek, *Comput. Appl. Biosci.* **10,** 41 (1994).
[46] H.-J. Bandelt and A. Dress, *Adv. Math.* **92,** 47 (1992).
[47] H.-J. Bandelt, *Verhandlungen des Naturwissenschaftlichen Vereins in Hambrug* **34,** 51 (1994).

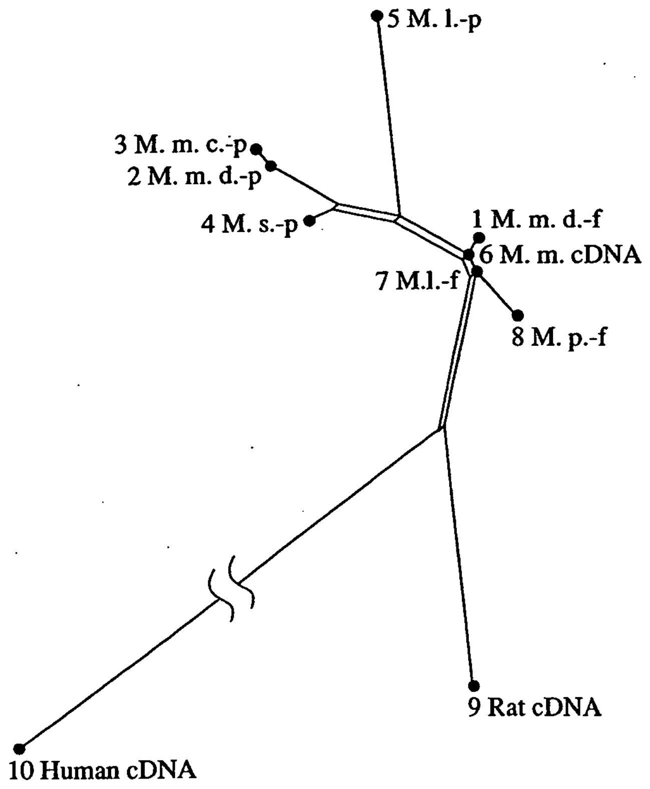


FIG. 9. Network constructed by using the SplitsTree program for the distance matrix of Table II. The length of an external branch to sequence 10 (human cDNA) was truncated, but other branch lengths were drawn proportional to the estimated lengths.

the PHYLIP WWW home page (http://evolution.genetics.washington.edu/phylip.html).

MEGA[28] is run on MS-DOS under a user-friendly environment. Many kinds of evolutionary distance estimation methods can be used, including synonymous and nonsynonymous substitutions. For further information, contact the following E-mail address: imeg@psuvm.psu.edu.

CLUSTAL W[29] is capable of doing multiple sequence alignment. After the alignment, it constructs neighbor-joining trees with bootstrapping. It can be retrieved from ftp.ebi.ac.uk (193.62.196.6) or from the EBI WWW home page (http://www.ebi.ac.uk/software/software.html).

MOLPHY[30] includes programs for maximum likelihood methods for both nucleotide and amino acid sequences. It can be retrieved via ftp from sunmh.ism.ac.jp (133.58.12.20).

Dendro-Maker (developed by Tadashi Imanishi) is run on Macintosh, and draws UPGMA and neighbor-joining trees. It can be retrieved via ftp from ftp.nig.ac.jp:/pub/mac/bio/dendromaker/. Treetool (developed by Mike Maciukenas) is run on Sun Sparc workstations and works with Newick format tree files for drawing trees. It can be retrieved through the RDP WWW homepage (http://rdp.life.uiuc.edu/).

TreeTree is a package of various programs mainly related to the neighbor-joining method developed by the author (E-mail address: nsaitou@genes.nig.ac.jp). Program NJ requires a distance matrix, whereas NJNUC requires nucleotide sequences. It can be retrieved through the author's WWW home page (http://smiler.nig.ac.jp/).

Acknowledgment

This chapter was partly supported by grants-in-aid for scientific researches from the Ministry of Education, Science and Culture, Japan.

[26] Estimating Evolutionary Distances between DNA Sequences

By WEN-HSIUNG LI and XUN GU

Introduction

Estimation of the evolutionary distance between two DNA sequences requires a stochastic model for nucleotide substitution. Most models for DNA evolution can be regarded as a time-continuous Markovian process, which can be characterized by the rate matrix **R**, or, equivalently, the nucleotide substitution pattern. The most general model for **R** has 12 parameters to be estimated, but its application in practice is difficult. Indeed, one usually uses a simpler model (i.e., a simplified substitution pattern) to derive an analytical formula for estimating the distance (see, e.g., Refs. 1–4).

However, if some assumptions of a simple substitution model are violated, the estimate of a distance will be biased and will not increase linearly with time, that is, it will be nonadditive.[5] Additivity is a highly desirable property for evolutionary distances, because if it does not hold, all distance matrix methods of tree reconstruction may become statistically inconsistent, that is, may lead to an erroneous tree with a probability approaching 1 as

[1] T. H. Jukes and C. R. Cantor, *in* "Mammalian Protein Metabolism" (H. N. Munro, ed.), p. 21. Academic Press, New York, 1969.
[2] M. Kimura, *J. Mol. Evol.* **16,** 111 (1980).
[3] F. Tajima and M. Nei, *Mol. Biol. Evol.* **1,** 269 (1984).
[4] K. Tamura and M. Nei, *Mol. Biol. Evol.* **10,** 512 (1993).
[5] J. Felsenstein, *Annu. Rev. Genet.* **22,** 521 (1988).

the sequence length increases to infinity.[6] To avoid this problem, it is necessary to develop a method for estimating distance under a general model of nucleotide substitution. Moreover, nonadditivity can also be generated if the assumption of a uniform rate among nucleotide sites is violated.[7] Because rate variation among sites is a common phenomenon, it should be taken into account in estimating evolutionary distances. In this chapter, we discuss an additive distance measure under a general substitution model and extend the method to the case where the substitution rate varies among sites. We first present the theory and then the estimation procedure. We also discuss the dissimilarities between our method and that of Lanave *et al.*,[8] which is one of the most general models available.

Stationary and Time-Reversible Model

Let **R** be the rate matrix whose *ij*th element r_{ij} is the substitution rate from nucleotide i to nucleotide j, $i \neq j$ ($i, j = 1, 2, 3$, and 4); the diagonal elements are given by $r_{ii} = -\Sigma_{j \neq i}\, r_{ij}$. By the Markovian process, the matrix of transition probabilities for t time units is generally given by

$$\mathbf{P}(t) = e^{\mathbf{R}t} \tag{1}$$

where the *ij*th element of $\mathbf{P}(t)$ is $P_{ij}(t)$, the probability of transition from nucleotide i to nucleotide j after t evolutionary time units.

The nucleotide substitution model to be used is called the SR model, which assumes that the substitution process is stationary and reversible in time. Stationarity means that the expected nucleotide frequencies in the sequences do not change in time. We denote the equilibrium frequencies by π_i, $i = 1, 2, 3$, and 4 for A, T, C, and G, respectively. The time reversibility means that the relation

$$\pi_i r_{ij} = \pi_j r_{ji} \tag{2}$$

holds for any i and j. Thus, the SR model can be presented by the following matrix.

	A	T	C	G
A	r_{11}	$\pi_2 v_1$	$\pi_3 v_2$	$\pi_4 s_1$
T	$\pi_1 v_1$	r_{22}	$\pi_3 s_2$	$\pi_4 v_3$
C	$\pi_1 v_2$	$\pi_2 s_2$	r_{33}	$\pi_4 v_4$
G	$\pi_1 s_1$	$\pi_2 v_3$	$\pi_3 v_4$	r_{44}

[6] R. W. DeBry, *Mol. Biol. Evol.* **9,** 537 (1992).
[7] Y. Tateno, N. Takezaki, and M. Nei, *Mol. Biol. Evol.* **11,** 261 (1994).
[8] C. Lanave, G. Preparata, C. Saccone, and G. Serio, *J. Mol. Evol.* **20,** 86 (1984).

TABLE I
MODELS OF NUCLEOTIDE SUBSTITUTION

JC model

	A	T	C	G
A	$-3v$	v	v	v
T	v	$-3v$	v	v
C	v	v	$-3v$	v
G	v	v	v	$-3v$

K2P model

	A	T	C	G
A	$-(2v + s)$	v	v	s
T	v	$-(2v + s)$	s	v
C	v	s	$-(2v + s)$	v
G	s	v	v	$-(2v + s)$

TN model

	A	T	C	G
A	r_{11}	$\pi_2 v$	$\pi_3 v$	$\pi_4 v$
T	$\pi_1 v$	r_{22}	$\pi_3 v$	$\pi_4 v$
C	$\pi_1 v$	$\pi_2 v$	r_{33}	$\pi_4 v$
G	$\pi_1 v$	$\pi_2 v$	$\pi_3 v$	r_{44}

TmN model

	A	T	C	G
A	r_{11}	$\pi_2 v$	$\pi_3 v$	$\pi_4 s_1$
T	$\pi_1 v$	r_{22}	$\pi_3 s_2$	$\pi_4 v$
C	$\pi_1 v$	$\pi_2 s_2$	r_{33}	$\pi_4 v$
G	$\pi_1 s_1$	$\pi_2 v$	$\pi_3 v$	r_{44}

Because $r_{ii} = -\Sigma_{j \neq i} r_{ij}$, for example, $r_{11} = -(\pi_2 v_1 + \pi_3 v_2 + \pi_4 s_1)$, the SR model is a nine-parameter model. The SR model includes many models as special cases, for example, the Jukes and Cantor model (JC),[1] the Kimura two-parameter model (K2P),[2] the Tajima and Nei model (TN),[3] and the Tamura and Nei model (TmN)[4] (see Table I).

General Additive Distance Measure

Consider two sequences X and Y that have evolved from O, a common ancestor, t time units ago. The transition probability matrix in each of the two lineages is given by Eq. (1). By time reversibility, the substitution process from the common ancestor O to sequences X and Y is equivalent to that from X through O to Y (or from Y through O to X). The transition probability matrix from X to Y is given by

$$\mathbf{P}(2t) = e^{2t\mathbf{R}} \tag{3}$$

The number of substitutions per site (K) between two DNA sequences, which is a standard measure of evolutionary distance, is defined by

$$K = 2t\sum_{i=1}^{4} \pi_i \sum_{j\neq i} r_{ij} = -2t\sum_{i=1}^{4} \pi_i r_{ii} \tag{4}$$

because $r_{ii} = -\Sigma_{j\neq i}\, r_{ij}$. By spectral decomposition, the diagonal elements of $\mathbf{R}$ can be expressed as

$$r_{ii} = \sum_{k=1}^{4} u_{ik} v_{ki} \lambda_k \tag{5}$$

where λ_k ($k = 1, 2, 3$, and 4) is the ith eigenvalue of $\mathbf{R}$, one of which is zero, say $\lambda_4 = 0$; u_{ik} is the ikth element of the eigenmatrix $\mathbf{U}$; and v_{ki} is the kith element of matrix $\mathbf{V} = \mathbf{U}^{-1}$. By putting Eq. (5) into Eq. (4), we have

$$K = -2t\sum_{k=1}^{3} b_k \lambda_k \tag{6}$$

where the constants b_k ($k = 1, 2$, and 3) are defined by

$$b_k = \sum_{i=1}^{4} \pi_i u_{ik} v_{ki} \tag{7}$$

Under the SR model, it is difficult to derive an analytical formula for K because the eigenvalues of $\mathbf{R}$ cannot be expressed in analytical forms. However, we can solve this problem as follows. Let z_k be the kth eigenvalue of $\mathbf{P}(2t)$. By matrix theory, Eq. (3) implies that

$$z_k = e^{2t\lambda_k} \tag{8}$$

Because $\lambda_4 = 0$ and $z_4 = 1$, there are only three nontrivial equations in Eq. (8). Equation (3) also implies that $\mathbf{R}$ and $\mathbf{P}(2t)$ have the same eigenmatrix. Thus, the number of substitutions per site can be computed by

$$K = -\sum_{k=1}^{3} b_k \ln z_k \tag{9}$$

where z_k and b_k can be estimated from sequence data (see below).

The method for estimating K under the SR model can be extended to

TABLE II
CONSTANTS c_k IN GENERAL ADDITIVE DISTANCE UNDER SR [EQ. (10)] OR SRV [EQ. (24)] MODEL[a]

Distance	c_k ($k = 1, 2,$ and 3)
K	$\sum_{i=1}^{4} \pi_i u_{ik} v_{ki}$
A	$\sum_{i=1}^{4} \sum_{j \neq i \in Ts} \pi_i u_{ik} v_{kj}$
B	$\sum_{i=1}^{4} \sum_{j \neq i \in Tv} \pi_i u_{ik} v_{kj}$
D_{ij}	$\pi_i u_{ik} v_{kj}$
d_m	$1/4$, if $z_k = \max(z_1, z_2, z_3)$; 0, otherwise

[a] K is the number of substitutions per site; A is the number of transitional substitutions per site; B is the number of transversional substitutions per site; D_{ij} is the number of substitutions from nucleotides i to j per site; and d_m is the minimum distance defined by Eq. (26). The subscripts $j \neq i \in Ts$ and $j \neq i \in Tv$ mean that the differences between nucleotides i and j are transitional (Ts) and transversional (Tv), respectively.

a general additive (time-linear) distance d, which is defined by the linear combination of ln z_k, namely,

$$d = -\sum_{k=1}^{3} c_k \ln z_k \qquad (10)$$

for some constants c_k. For example, the number of transitional substitutions per site (A), the number of transversional substitutions per sites (B), and the number of substitutions from nucleotides i to j (D_{ij}) are the special cases of d by choosing appropriate constants c_k (see Table II).

However, the general additive distance defined by Eq. (10) requires the condition that all eigenvalues z_k (or λ_k) are real. We[9] have shown that this is the case.

Estimation of Distances and Sampling Variances

As all the above quantities (K, A, B, and D_{ij}) can be expressed in the same form (Table II), we can treat all of them in the same way. Let J_{ij} be

[9] X. Gu and W. H. Li, *Proc. Natl. Acad. Sci. U.S.A.*, in press (1996).

the expected frequency of sites where the nucleotide is i in sequence X and j in sequence Y, which, by Markovian properties, is given by

$$J_{ij} = \sum_{k=1}^{4} \pi_k P_{ki}(t) P_{kj}(t), \qquad i,j = 1, \ldots, 4 \tag{11}$$

By time reversibility, that is, $\pi_i P_{ij}(t) = \pi_j P_{ji}(t)$, we have

$$J_{ij} = \sum_{k=1}^{4} \pi_i P_{ik}(t) P_{kj}(t) = \pi_i \sum_{k=1}^{4} P_{ik}(t) P_{kj}(t) = \pi_i P_{ij}(2t) \tag{12}$$

where $\Sigma_{k=1}^{4} P_{ik}(t) P_{kj}(t) = P_{ij}(2t)$ is a basic property of transition probabilities. Obviously, Eq. (12) gives a simple method for estimating the transition probability $P_{ij}(t)$ directly from sequence data J_{ij}.

Let matrix **J** consist of J_{ij}. It can be shown that, if the substitution process is stationary and reversible, **J** is symmetric, that is, $J_{ij} = J_{ji}$. To test whether the observed data deviate significantly from this condition, we suggest the following χ^2 test. Let N_{ij} be the observed number of sites at which the nucleotide is i in sequence X and j in sequence Y. Note that there are six independent equations if **J** is symmetric, namely, $E[N_{12}] = E[N_{21}]$, $E[N_{13}] = E[N_{31}]$ and so forth, where E means taking expectation. For each pair, say $E[N_{12}]$ versus $E[N_{21}]$, we can construct the statistic $(N_{12} - N_{21})^2/(N_{12} + N_{21})$ to test whether the observed data deviate significantly from the null hypothesis of $E[N_{12}] = E[N_{21}]$, which follows a χ^2 distribution with $df = 1$. Therefore, the following statistic

$$S = \sum_{i=1}^{3} \sum_{j=i+1}^{4} \frac{(N_{ij} - N_{ji})^2}{N_{ij} + N_{ji}} \tag{13}$$

follows a χ^2 distribution with $df = 6$. Thus, if $S > 12.59$, the null hypothesis that $N_{ij} = N_{ji}$ for all $i \neq j$ is rejected at the 5% significance level; in this case, the SR model cannot be applied.

In summary, the procedure for estimating distances can be outlined as follows (a computer program for the entire procedure is available on request).

Step 1. For each pair of sequences, count N_{ij}, the number of sites at which the nucleotide is i in the first sequence and is j in the second sequence.

Step 2. Compute the statistic S given by Eq. (13) to test whether matrix **J** is symmetric. Stop if **J** is nonsymmetric, that is, if $S \geq 12.59$.

Step 3. Estimate matrix **J** by

$$\hat{J}_{ij} = \frac{N_{ij} + N_{ji}}{2L}, \qquad i,j = 1, \ldots, 4 \tag{14}$$

where L is the sequence length.

Step 4. Estimate the transition probability matrix $\mathbf{P}(2t)$ as

$$\hat{P}_{ij} = \frac{\hat{J}_{ij}}{\hat{\pi}_i}, \qquad i, \ldots, 4 \tag{15}$$

where $\hat{\pi}_i$ is the frequency of nucleotide i estimated by taking (simple) average between the two sequences.

Step 5. Compute eigenvalues $\hat{z}_k$ ($k = 1, \ldots, 4$) by solving the characteristic equation $\det(\hat{\mathbf{P}} - z\mathbf{I}) = 0$, where $\hat{\mathbf{P}}$ consists of $\hat{P}_{ij}$ and $\mathbf{I}$ is the identity matrix; the corresponding eigenmatrix $\mathbf{U}$ and its inverse matrix $\mathbf{V}$ are also obtained simultaneously by a standard algorithm.

Step 6. Compute the evolutionary distance d from Eq. (10); the constants c_i depend on the specified distance measure (see Table II).

Step 7. Finally, compute the sampling variance of d approximately given by

$$\operatorname{Var}(d) = \sum_{i=1}^{3} \frac{c_i^2}{z_i^2} \operatorname{Var}(z_i) + \sum_{i=1}^{3} \sum_{j \neq i} \frac{c_i c_j}{z_i z_j} \operatorname{Cov}(z_i, z_j) \tag{16}$$

where $\operatorname{Var}(z_i)$ is the sampling variance of z_i and $\operatorname{Cov}(z_i, z_j)$ is the sampling covariance between z_i and z_j. Let $\mathbf{W}$ be the variance–covariance matrix of z_i values, which is defined by $w_{ii} = \operatorname{Var}(z_i)$ and $w_{ij} = \operatorname{Cov}(z_i, z_j)$ ($i \neq j$). We[9] have developed a simple method to compute $\mathbf{W}$ by inverting $\mathbf{I}_f$, that is, $\mathbf{W} = \mathbf{I}_f^{-1}$, where the klth element of matrix $\mathbf{I}_f$ is given by

$$I_{kl} = \sum_{i=1}^{4} \sum_{j=1}^{4} \frac{N_{ij}}{P_{ij}^2} u_{ik} u_{il} v_{kj} v_{lj}, \qquad k, l = 1, 2, 3 \tag{17}$$

General Additive Distance Under Variable Rates

The general additive distance defined by Eq. (10) requires the assumption of the same substitution rate for all sites. If this assumption is violated, Eq. (10) may no longer be additive.[7] Thus, we need to extend the new method to the case where the substitution rate varies among sites (the SRV model, where V stands for variable rates).

We assume that the rate variation among sites follows a gamma distribution. The model is as follows. The ijth element of the rate matrix $\mathbf{R}$ is expressed by $r_{ij} = h_{ij} u$, where h_{ij} is a constant and u varies randomly according to the following gamma distribution

$$\phi(u) = \frac{\beta^\alpha}{\Gamma(\alpha)} u^{\alpha - 1} e^{-\beta u} \tag{18}$$

where the mean of u is given by $\bar{u} = \alpha/\beta$.

First, we consider the expected number of substitutions per site K, which, under the SRV model, is defined by

$$K = -2t \sum_{i=1}^{4} \pi_i \bar{r}_{ii} \tag{19}$$

where $\bar{r}_{ij} = h_{ij}\bar{u}$. Let $\bar{\mathbf{R}}$ be the matrix of expected rates, that is, $\bar{\mathbf{R}}$ consists of $\bar{r}_{ii}$. By spectral decomposition, Eq. (19) can be written as

$$K = -2t \sum_{k=1}^{3} b_k \bar{\lambda}_k \tag{20}$$

where constants b_k are given by Eq. (7); $\bar{\lambda}_k$ ($k = 1, \ldots, 4$) is the kth eigenvalue of matrix $\bar{\mathbf{R}}$ ($\bar{\lambda}_4 = 0$).

Note that, when the rate varies among sites, only the expectation (over all sites) can be observed from sequence data. That is, the $\hat{P}_{ij}$ of Eq. (15) is actually an estimate of the average transition probability $\bar{P}_{ij}(2t)$, which is defined by

$$\bar{P}_{ij}(2t) = \int_0^{\infty} P_{ij}(2t)\phi(u)\, du \tag{21}$$

Let $\bar{\mathbf{P}}(2t)$ be the matrix of average transition probabilities, and $\bar{z}_k$ ($k = 1, \ldots, 4$) be the kth eigenvalues of matrix $\bar{\mathbf{P}}(2t)$. To derive an estimation formula of K under the SRV model, one must find a relation between $\bar{\lambda}_k$ and $\bar{z}_k$. We[9] have shown that, when the substitution rate varies among sites according to a gamma distribution, $\bar{z}_k$ and $\bar{\lambda}_k$ have the following relation

$$-2\bar{\lambda}_k t = \alpha(\bar{z}_k^{-1/\alpha} - 1), \qquad i = 1, \ldots, 4 \tag{22}$$

The expected number of substitutions per site under the SRV model is given by

$$K = \alpha \sum_{k=1}^{3} b_k(\bar{z}_k^{-1/\alpha} - 1) \tag{23}$$

and the general additive distance under the SRV model is given by

$$\bar{d} = \alpha \sum_{k=1}^{3} c_k(\bar{z}_k^{-1/\alpha} - 1) \tag{24}$$

for some constants c_k (see Table II). It can be shown that $\bar{d} \rightarrow d$ of Eq. (10) when $\alpha \rightarrow \infty$, that is, the substitution rate is uniform among sites.

The procedures for estimating distance $\bar{d}$ under the SRV model are the same as that of the SR model except the last two steps, that is, after step 5, go to the following steps:

Step 6′. Compute the evolutionary distance $\bar{d}$ by Eq. (24) if the gamma distribution parameter α is known; it can be estimated by the parsimony method[4] or the maximum likelihood method[10]; the constants c_i depend on the specified distance measure (see Table II).

Step 7′. Finally, compute the sampling variance of $\bar{d}$ approximately given by

$$\mathrm{Var}(\bar{d}) = \sum_{i=1}^{3} \frac{c_i^2}{z_i^{2(1+1/\alpha)}} \mathrm{Var}(z_i) + \sum_{i=1}^{3} \sum_{j\neq i} \frac{c_i c_j}{(z_i z_j)^{1+1/\alpha}} \mathrm{Cov}(z_i, z_j) \tag{25}$$

Obviously, $\mathrm{Var}(\bar{d}) \to \mathrm{Var}(d)$ as $\alpha \to \infty$. $\mathrm{Var}(\bar{d})$ in Eq. (25) should be interpreted as a lower bound of the sampling variance because it assumes that α is estimated precisely without error.

Minimum Distance and Inapplicable Cases

When the sequence length is short, the sampling effect can become serious. The large sampling variance can not only nullify the additivity of a distance, but can also make the SR method inapplicable. An inapplicable case occurs if any estimated z_k is negative. Therefore, a distance measure that minimizes the sampling effects may be useful in practice. Let $z_m = \max(z_1, z_2, z_3)$. Then the minimum distance is defined by

$$d_m = -\frac{1}{4} \ln z_m \tag{26}$$

The sampling variance of d_m is given by

$$\mathrm{Var}(d_m) = \left(\frac{1}{4z_m}\right)^2 \mathrm{Var}(z_m) \tag{27}$$

where $\mathrm{Var}(z_m)$ can be computed as above. Note that d_m is always positive because $0 < z_1, z_2, z_3 < 1$ under the SR model. The minimum distance is additive and less affected by saturation so that it has few inapplicable cases. Note that d_m is more general than the transversion distance. Indeed, under the Tamura and Nei model,[4] which is a special case of the SR model, the

[10] X. Gu, X. Y. Fu, and W. H. Li, *Mol. Biol. Evol.* **12,** 546 (1995).

transversional distance is the minimum distance d_m if the rates of transversion are smaller than those of transition.

However, as shown by Kelly,[11] negative eigenvalues can arise if the nucleotide substitution does not follow a Markov model. We can assess whether the substitution process fits the general Markovian model, using a test that the smallest eigenvalue is positive. We suggest a Z test by computing the following statistic

$$Z = \frac{z_0}{\text{Var}(z_0)^{1/2}} \tag{28}$$

where $z_0 = \min(z_1, z_2, z_3, z_4)$. Note that in the case of $z_0 < 0$, the SR method is inapplicable. If z_0 is not significantly smaller than zero, the inapplicability may be regarded as due to sampling effects, and the minimum distance or a simpler method may be used instead. However, if z_0 is significantly smaller than 0, the substitution process may not be Markovian; in this case, a more complex model is needed.

Discussion

The model of Lanave *et al.*[8] (LA for short) has been shown to be a nine-parameter rather than twelve-parameter model.[12] So, it is in fact equivalent to the SR model. However, their method and ours differ in several aspects. First, our additive distance is general and includes many distance measures as special cases; the LA method considers only the number of substitutions per site. Second, our method uses a χ^2 test to assess whether the SR model is suitable; the test in the LA method is for testing the stationarity but not time reversibility. Third, and most important, we extended the general additive distance under the SR model to the case where the substitution rate varies among sites (i.e., the SRV model); there is no similar extension in the LA method. Therefore, our method has several advantages over the LA method.

Rodriguez *et al.*[13] proposed a general formula for estimating the number of substitutions per site as

$$K = -tr[\mathbf{F} \ln(\mathbf{F}^{-1}\mathbf{J})] \tag{29}$$

[11] C. Kelly, *Biometrics* **50,** 653 (1994).
[12] A. Zharkikh, *J. Mol. Evol.* **39,** 315 (1994).
[13] F. Rodriguez, J. F. Oliver, A. Marin, and J. R. Medina, *J. Theor. Biol.* **142,** 485 (1990).

where tr is the trace of a matrix and **F** is the diagonal matrix of nucleotide frequencies, that is, $\mathbf{F} = \text{diag}(\pi_1, \pi_2, \pi_3, \pi_4)$. It can be shown that Eq. (29) is mathematically equivalent to Eq. (9). However, Eq. (29) is not convenient for extension to include the effect of rate heterogeneity. The general additive distance defined by Eq. (10) is more general and easier for conducting statistical analysis of sequence data.

It is worth mentioning that if $c_i = 1/4$, Eq. (10) is the paralinear or LogDet distance.[14–16] Let matrix **J** consist of J_{ij}. Then, Eq. (12) can be expressed as $\mathbf{J} = \mathbf{FP}$, or $\mathbf{P} = \mathbf{F}^{-1}\mathbf{J}$. Because $z_1z_2z_3z_4 = \det(\mathbf{P}) = \det(\mathbf{F}^{-1}\mathbf{J}) = \det(\mathbf{J})/\det(\mathbf{F})$, we have

$$d = -\frac{1}{4}\ln(z_1z_2z_3z_4) = -\frac{1}{4}\ln\frac{\det(\mathbf{J})}{\det(\mathbf{F})} \tag{30}$$

The paralinear distance was recommended for phylogenetic reconstruction because its additivity holds even when the process is not reversible in time, that is, it is based on the general (12-parameter) substitution model. However, the paralinear distance is an unbiased estimate of the number of substitutions per site (K) if and only if the equilibrium frequencies of the four nucleotides are 1/4. More seriously, the additivity does not hold if the substitution rate varies among sites. To our knowledge, the SRV method is to date the most general that includes the effect of rate variation among sites. Since rate variation is a common phenomenon, the SRV method is preferable over the paralinear distance.

Acknowledgments

We thank Dr. A. Zharkikh for valuable discussion. This study was supported by National Institutes of Health Grant GM30998.

[14] D. Barry and J. A. Hartigan, *Biometrics* **43,** 261 (1987).
[15] J. A. Lake, *Proc. Natl. Acad. Sci. U.S.A.* **91,** 1455 (1994).
[16] P. J. Lockhart, M. A. Steel, M. D. Hendy, and D. Penny, *Mol. Biol. Evol.* **11,** 605 (1994).

[27] Local Alignment Statistics

By STEPHEN F. ALTSCHUL and WARREN GISH

Introduction

Because protein and DNA sequences frequently share only isolated regions of similarity, the most widely used sequence alignment algorithms seek subalignments[1-3] as opposed to global alignments.[4-7] A subalignment (also called a local alignment) places into correspondence a segment from each of the two sequences being compared. Null characters (often represented by dashes) may be added to either segment to represent insertions or deletions. Subalignment quality generally is measured by a score calculated by adding substitution scores for each aligned pair of letters and gap scores for each run of nulls in one segment aligned with letters in the other. An ungapped subalignment is one in which no nulls or gaps are allowed.

Given a particular scoring system, a central question has always been how high a score may be expected to occur purely by chance. Through studying the comparison of random sequences, much progress has been made on this question. The optimal scores from ungapped subalignments are known to approach an extreme value distribution, and explicit formulas are available for the parameters of the distribution.[8-11] The distribution for the sum of the r best ungapped subalignment scores is also known.[12] Some rougher asymptotic results are available for the scores of subalignments with gaps,[13] and there is much empirical evidence that the statistical theory for ungapped subalignments generalizes in outline to this case.[14-18]

[1] T. F. Smith and M. S. Waterman, *J. Mol. Biol.* **147,** 195 (1981).
[2] W. R. Pearson and D. J. Lipman, *Proc. Natl. Acad. Sci. U.S.A.* **85,** 2444 (1988).
[3] S. F. Altschul, W. Gish, W. Miller, E. W. Myers, and D. J. Lipman, *J. Mol. Biol.* **215,** 403 (1990).
[4] S. B. Needleman and C. D. Wunsch, *J. Mol. Biol.* **48,** 443 (1970).
[5] D. Sankoff, *Proc. Natl. Acad. Sci. U.S.A.* **69,** 4 (1972).
[6] P. H. Sellers, *SIAM J. Appl. Math.* **26,** 787 (1974).
[7] D. Sankoff and J. B. Kruskal, "Time Warps, String Edits and Macromolecules: The Theory and Practice of Sequence Comparison." Addison-Wesley, Reading, Massachusetts, 1983.
[8] S. Karlin and S. F. Altschul, *Proc. Natl. Acad. Sci. U.S.A.* **87,** 2264 (1990).
[9] A. Dembo, S. Karlin, and O. Zeitouni, *Ann. Prob.* **22,** 2022 (1994).
[10] A. Dembo and S. Karlin, *Ann. Prob.* **19,** 1737 (1991).
[11] A. Dembo and S. Karlin, *Ann. Prob.* **19,** 1756 (1991).
[12] S. Karlin and S. F. Altschul, *Proc. Natl. Acad. Sci. U.S.A.* **90,** 5873 (1993).
[13] R. Arratia and M. S. Waterman, *Ann. Appl. Prob.* **4,** 200 (1994).
[14] T. F. Smith, M. S. Waterman, and C. Burks, *Nucleic Acids Res.* **13,** 645 (1985).

In this chapter, we study further the distribution of optimal gapped subalignment scores. We provide evidence that two parameters are sufficient to describe both the form of this distribution and its dependence on sequence length. Using a random protein model, the relevant statistical parameters are calculated for a variety of substitution matrices and gap costs. An analysis of these parameters elucidates the relative effectiveness of affine as opposed to length-proportional gap costs. We show empirically that the theory for the sum of the r best ungapped subalignment scores generalizes to gapped subalignments. Thus sum statistics provide a method for evaluating sequence similarity that treats short and long gaps differently. By example, we show how this method has the potential to increase search sensitivity. The statistics described can be applied to the results of FASTA[2] searches, or to those from a variation of the BLAST programs, described here, that permits gaps.

Statistics of Ungapped Subalignment Scores

The scores of ungapped subalignments are inherited completely from scores for aligning pairs of residues. The score for aligning residues i and j we will call s_{ij}, and we assume generally that $s_{ij} = s_{ji}$. To approach analytically the question of what subalignment scores are statistically significant, a model of random sequences is needed. The simplest model assumes that all residues are drawn independently, with respective probabilities $p_1, \ldots, p_r$ for the various residue types; r is 20 for proteins and 4 for nucleic acids. For the statistical theory described here to hold, a central requirement is that the expected score $\Sigma_{i,j=1}^{r} p_i p_j s_{ij}$ for a pair of randomly chosen residues be negative. This is, in fact, a desirable restriction, for a positive expected score implies that long subalignments will tend to have high scores, even when the constituent segments bear no biological relationship.

Let S be the optimal ungapped subalignment score from the comparison of two random sequences of lengths m and n. When m and n are sufficiently large, it can be shown that the distribution of S is well approximated by an extreme value distribution,[19] whose cumulative distribution function is given by

$$P(S < x) = \exp[-e^{-\lambda(x-u)}] \qquad (1)$$

[15] J. F. Collins, A. F. W. Coulson, and A. Lyall, *CABIOS* **4,** 67 (1988).
[16] R. Mott, *Bull. Math. Biol.* **54,** 59 (1992).
[17] M. S. Waterman and M. Vingron, *Proc. Natl. Acad. Sci. U.S.A.* **91,** 4625 (1994).
[18] M. S. Waterman and M. Vingron, *Stat. Sci.* **9,** 367 (1994).
[19] E. J. Gumbel, "Statistics of Extremes." Columbia Univ. Press, New York, 1958.

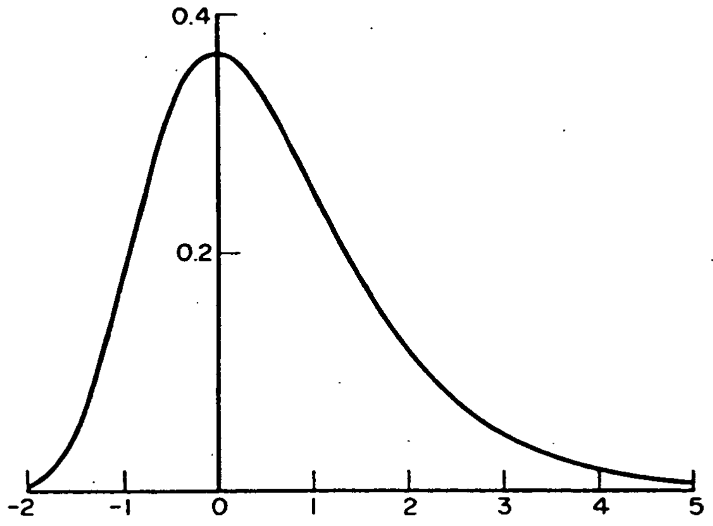


FIG. 1. Probability density function for the extreme value distribution with $u = 0$ and $\lambda = 1$.

This distribution has two parameters: the characteristic value u, which can be thought of as the center of the distribution, and the decay constant or scale parameter λ. The probability density function for the standard extreme value distribution with $u = 0$ and $\lambda = 1$ is shown in Fig. 1.

Analytic formulas are available for λ and u in the case considered here.[8-10] λ is the unique positive solution for x in Eq. (2):

$$\sum_{i,j=1}^{r} p_i p_j e^{s_{ij}x} = 1 \tag{2}$$

u is dependent on the lengths of the sequences compared and is given by

$$u = (\ln Kmn)/\lambda \tag{3}$$

where K is a constant given by a geometrically convergent series dependent on the p_i and s_{ij}.[8-10] Combining Eqs. (1) and (3), we can eliminate u and write simply that

$$P(S \geq x) = 1 - \exp(-Kmn\ e^{-\lambda x}) \tag{4}$$

This is an expression for the probability that the optimal ungapped subalignment attains a score of at least x. More generally, it may be shown that the number of distinct ungapped subalignments with score at least x is approximately Poisson distributed,[8] with parameter $Kmn\ e^{-\lambda x}$. The meaning of "distinct" will be discussed below at greater length. These results may be generalized to Markov-dependent sequences[11] and to ungapped subalignments of multiple sequences.

We note that when a set of scores s_{ij} is multiplied by a positive scalar, the effect is to divide λ by that same constant. Because the new scores differ from the original ones in no important way (they both imply the same optimal subalignments), it is useful to define normalized scores, which are λ times the nominal scores, that remain fixed. These scores are said to be expressed in nats,[20] and we use this terminology below. When P of Eq. (4) is less than 0.1, increasing the score of a subalignment by 1 nat decreases its probability of chance occurrence by a factor of about e.

Dependence of Statistical Parameters on Search Space Size

An extreme value distribution has been shown to hold in the asymptotic limit for other measures of local similarity, such as the longest run of matching letters between two sequences, allowing for k mismatches.[21] As above, explicit formulas for λ and u are available. If p is the probability that two random letters match, $\lambda = \ln 1/p$, and u is a function of $(1 - p)mn$ involving constant, log, and log–log terms.[21]

The measure of sequence similarity S of interest to us here is the optimal subalignment score, allowing gaps, frequently called a Smith–Waterman score.[1] We will consider only affine gap costs,[22,23] parameterized by a gap-opening penalty a, and a gap-extension penalty b, so that a gap of length k receives a score of $-a - (k - 1)b$. It has been shown that for a range of substitution and gap scores, S grows asymptotically as a logarithmic function of the search space size mn.[13] The scores for which this is true are analogous to scores with negative expected value in the no-gap case, and it is only in this local scoring regime that we will be interested.

Although the distribution of S has not been shown analytically to approach an extreme value distribution, a number of empirical studies strongly suggest that it does.[14–18] To obtain estimates of statistical significance under this assumption, one need only estimate the relevant parameters λ and u for use in Eq. (1). These parameters will, in general, depend on the particular scoring system employed, and on the compositions and lengths of the sequences being compared.

If a particular protein scoring system has been chosen, and proteins can be assumed to have a standard amino acid composition, then λ and u will be a function only of the lengths of the sequences being compared. There is much empirical evidence that λ remains unchanged with increasing search

[20] S. F. Altschul, *J. Mol. Biol.* **219,** 555 (1991).
[21] R. Arratia, L. Gordon, and M. S. Waterman, *Ann. Stat.* **14,** 971 (1986).
[22] O. Gotoh, *J. Mol. Biol.* **162,** 705 (1982).
[23] W. M. Fitch and T. F. Smith, *Proc. Natl. Acad. Sci. U.S.A.* **80,** 1382 (1983).

space size, and this agrees with the analytic results for simpler measures of local similarity. It would be convenient if, as hypothesized by Waterman and Vingron,[18] the dependence of u on search space size takes the simple form of Eq. (3) that it does when gaps are disallowed: a simulation for one search space size could then be generalized to all sizes. It is not immediately clear, however, that this should be the case. We know, for instance, that for the longest matching run with k mismatches, the formula for u contains a log–log term in mn,[21] and there is empirical evidence that such a term arises as well for various other local similarity functions.[24–26] An earlier empirical study finds evidence of a log–log term for Smith–Waterman scores,[16] and Waterman and Vingron find some deviation from their estimated probability curves when mn is varied.[18] We show in the next section that a naive data analysis indeed suggests the presence of a log–log term in the formula for u, but that a simple "edge effect" correction eliminates the evidence for such a term. Thus it appears that the simple form of the theory for ungapped subalignments does extend to gapped subalignments.

Estimation of Statistical Parameters

The most straightforward way to estimate λ and u for the comparison of sequences of lengths m and n is to generate many pairs of random sequences and calculate for each the optimal subalignment score S. The parameters for an extreme value distribution may be fit to these data using the method of moments,[24] maximum likelihood estimation,[16] or linear regression on a transform of the data[15,17,18]; all these methods give virtually identical results. A rapid method for estimating the parameters, which involves collecting scores from the r locally best subalignments, has also been described.[17,18] In the present study, λ and u are estimated by the method of moments.

Using the background amino acid frequencies described by Robinson and Robinson,[27] we generated 10,000 pairs of random protein sequences for each of a large range of sequence lengths $m = n$. For each pair, we calculated the optimal subalignment score based on the BLOSUM62 substitution scores,[28] coupled with ($a = 12$, $b = 1$) affine gap costs. For reasons discussed below, we collected as well the length of each optimal subalignment. As in other studies,[16–18] the resulting scores fit well an extreme value

[24] S. F. Altschul and B. W. Erickson, *Bull. Math. Biol.* **48,** 617 (1986).

[25] M. S. Waterman and L. Gordon, *in* "Computers and DNA" (G. I. Bell and T. G. Marr, eds.), p. 127. Addison-Wesley, Reading, Massachusetts, 1990.

[26] S. F. Altschul, *J. Mol. Evol.* **36,** 290 (1993).

[27] A. B. Robinson and L. R. Robinson, *Proc. Natl. Acad. Sci. U.S.A.* **88,** 8880 (1991).

[28] S. Henikoff and J. G. Henikoff, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 10915 (1992).

TABLE I
EMPIRICAL VALUES FOR l, u, λ, AND K AS FUNCTION OF SEARCH SPACE SIZE[a]

n, m	l	ln nm	ln $n'm'$	u	λ	K
191	22.6	10.5	10.25	26.45	0.298	0.073
245	25.8	11.0	10.78	28.31	0.286	0.055
314	29.4	11.5	11.30	30.21	0.282	0.051
403	32.4	12.0	11.83	32.04	0.275	0.041
518	36.3	12.5	12.35	33.92	0.279	0.048
665	40.3	13.0	12.87	35.94	0.273	0.041
854	43.9	13.5	13.39	37.84	0.272	0.040
1097	48.1	14.0	13.91	39.75	0.275	0.046
1408	51.6	14.5	14.43	41.71	0.268	0.036
1808	55.1	15.0	14.94	43.54	0.271	0.041
2322	59.1	15.5	15.45	45.53	0.267	0.035
2981	63.5	16.0	15.96	47.32	0.270	0.040

[a] Ten thousand random protein sequence pairs, using amino acid frequencies from Robinson and Robinson,[27] were generated for each value of n and m. Optimal subalignment scores were calculated using (12, 1) affine gap costs and the BLOSUM62 amino acid substitution matrix.[28] Standard errors for u and λ are 0.05 and 0.003, respectively.

distribution. We confine our attention here to the attendant estimates of λ and u, shown in Table I. For further analysis the table also provides estimates of the parameter K, under the assumption that Eq. (3) is valid.

As expected,[14–18] the estimates of λ remain essentially constant for m and n sufficiently large (>350). We turn thus to the question of whether the parameter u can in fact be written in the form of Eq. (3), or whether a log–log term is suggested as well. Equation (3) implies that plotting u against ln mn should yield a straight line with slope $1/\lambda$ and y intercept $(\ln K)/\lambda$. Linear regression of u versus ln mn indeed yields an almost perfectly straight line ($r > 0.9999$), but the implied value of 0.261 for λ is significantly lower than the average value of 0.272 found for those points with m and n greater than 350. The greater than anticipated slope could be explained by a log–log term, as invoked by Mott.[16] Although the 4% discrepancy in λ may seem small, a similar experiment using PAM250[29,30]

[29] M. O. Dayhoff, R. M. Schwartz, and B. C. Orcutt, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 345. National Biomedical Research Foundation, Washington, D.C., 1978.

[30] R. M. Schwartz and M. O. Dayhoff, *in* "Atlas of Protein Sequence and Structure" (M. O. Dayhoff, ed.), Vol. 5, Suppl. 3, p. 353. National Biomedical Research Foundation, Washington, D.C., 1978.

TABLE II
STATISTICAL PARAMETERS CALCULATED BY VARIOUS METHODS

Scoring system	Method for calculating parameters	λ	K
BLOSUM62 matrix, (12, 1) gap costs	Average case ($n, m > 350$)	0.272	0.041
	Linear regression	0.261	0.026
	Edge-corrected linear regression	0.270	0.041
PAM250 matrix, (15, 3) gap costs	Average case ($n, m > 350$)	0.202	0.037
	Linear regression	0.189	0.019
	Edge-corrected linear regression	0.199	0.037
BLOSUM62 matrix, infinite gap costs	Average case ($n, m > 200$)	0.321	0.12
	Linear regression	0.311	0.08
	Edge-corrected linear regression	0.319	0.12
	Analytic theory	0.318	0.13

substitution scores and (15, 3) gap costs yielded an even larger discrepancy of 6% (see Table II).

Before postulating the necessity for a more complicated formula than Eq. (3), we tested an edge effect correction of a type that has been used for many years by the BLAST programs.[3] The basic idea is that a subalignment does not exist at a point, but has a certain length. Any subalignment starting near the end of either sequence is thus likely to run out of sequence before it can attain a score sufficient to become optimal. A sequence might therefore be considered to have an effective length shorter than its actual one by the length l of a typical optimal subalignment. Accordingly, we provide in Table I a column l representing the mean of the observed lengths of the optimal subalignments, and a column $\ln m'n'$, where $m' = m - l$ and $n' = n - l$. Performing a linear regression using these edge-corrected values for search space size yields a value of 0.270 for λ, in very close agreement with the average value of λ estimated from individual points. The same improved agreement was found for the PAM250-based scoring system described above (Table II).

To test the validity of this approach, we performed a similar computational experiment for a case in which the asymptotic values for λ and K are known analytically, that is, one in which no gaps are allowed. As shown in Table II, the parameter estimates both from edge-corrected linear regression and from averaging the estimates from multiple values of m and n agree quite well with the values of λ and K derived from theory.

Although no computational experiments are sufficient to establish the asymptotic validity of a specific formula for u, or indeed the validity of the extreme value distribution, our results certainly are consistent with the hypothesis of Waterman and Vingron[18] that u can be well approximated by Eq. (3). This is fortunate, for it allows a simulation for a single pair of sequence lengths to determine K, and thereby to determine u for all other search space sizes. Our one suggestion is that, in addition to parameters λ and K, the relative entropy H[20] of the scoring system should be estimated. Basically, H is the expected score, per aligned pair of residues, of an optimal random subalignment. By estimating the mean length l described above, H expressed in nats[28] may be written as

$$H \approx \lambda u / l \approx (\ln Kmn)/l \tag{5}$$

This formula may then be inverted to calculate the edge correction described above. Specifically, $m' \approx m - (\ln Kmn)/H$, and similarly for n'. When $(\ln Kmn)/H$ is a large fraction of either m or n, it suggests that edge effects are substantial, and that asymptotic formulas are likely to lose some accuracy.

Local Optimality

Two sequences may share more than a single region of similarity, and it is therefore desirable to seek not just the optimal subalignment, but other high-scoring subalignments as well. The immediate problem is that the second, third, and fourth highest scoring subalignments are all likely to be slight variations of the optimal one. A definition is needed of when two subalignments are distinct.

The first such definition is due to Sellers,[31] who described a subalignment as locally optimal if it has a better score than any subalignment it intersects within a path graph. He provided an algorithm to find all and only the locally optimal subalignments,[31] but it required an undetermined number of sweeps of a path graph, and therefore was not provably $O(n^2)$. An $O(n^2)$ time algorithm was subsequently described.[32]

Altschul and Erickson introduced a definition of weak local optimality, somewhat more inclusive than the Sellers original definition.[32] A subalignment is weakly locally optimal if it intersects within a path graph no better subalignment that is weakly locally optimal. This definition is not circular but recursive; it is anchored by the optimal subalignment. One problem with seeking only locally optimal subalignments is that a very strong subalignment A can suppress a good subalignment C it never intersects by means of a subalignment B, with intermediate score, that intersects both.[24,32] By the Sellers definition, only subalignment A could be locally optimal; by the definition of Altschul and Erickson, both A and C could be weakly

[31] P. H. Sellers, *Bull. Math. Biol.* **46,** 501 (1984).
[32] S. F. Altschul and B. W. Erickson, *Bull. Math. Biol.* **48,** 633 (1986).

locally optimal. From a mathematical standpoint local optimality is the more natural definition, but weak local optimality generally captures better most people's intuitive notion of when two subalignments are distinct.

Algorithms for finding all weakly locally optimal subalignments using nonlinear similarity functions were described by Altschul and Erickson.[24,32] Waterman and Eggert described a more efficient algorithm for the standard linear similarity function under consideration here.[33] For real biological sequences the Waterman–Eggert algorithm generally requires only $O(n^2)$ time, but its worst case behavior has not been established. A linear-space variation of this algorithm has been described by Huang *et al.*[34] Barton has described an algorithm claimed to be more time efficient than that of Waterman and Eggert.[35] Although the algorithm is more efficient, it does not in fact always succeed in finding all and only the locally optimal subalignments by either definition above,[36,37] and it must therefore be regarded as a heuristic.

In the next section we consider the random behavior of the r best locally optimal subalignments. We use the Sellers definition of local optimality because it permits a provably $O(n^2)$ algorithm[32] that is simpler than that of Waterman and Eggert for weakly locally optimal subalignments. We note that for random sequences, where the optimal subalignment never has a very large score, and where locally optimal subalignments tend to be far removed from one another, the two definitions usually are congruent. Thus our results should apply both to locally optimal subalignments and to weakly locally optimal subalignments. For the reasons discussed above, however, when real sequences are to be compared we recommend the weak local optimality definition,[32] and thus the algorithms of Waterman and Eggert[33] or Huang *et al.*[34]

Sum Statistics

When several distinct regions of similarity are found within a pair of sequences, it is sometimes desirable to report a combined assessment of their statistical significance. This is more involved, however, than it might first appear. Suppose, for example, that using a particular scoring system the two best distinct subalignments have scores of 65 and 40. It is tempting to ask for the probability that the highest score is 65 or greater and that

[33] M. S. Waterman and M. Eggert, *J. Mol. Biol.* **197,** 723 (1987).
[34] X. Huang, R. C. Hardison, and W. Miller, *CABIOS* **6,** 373 (1990).
[35] G. J. Barton, *CABIOS* **9,** 729 (1993).
[36] O. Gotoh, *CABIOS* **3,** 17 (1987).
[37] W. Miller and M. Boguski, *CABIOS* **10,** 455 (1994).

the second highest score is 40 or greater. This, however, is inappropriate, because a legitimate p value represents the probability of attaining a given result or a better one, and thus requires a linear ranking of all possible results. Call the above result A, and consider a different result B in which the highest score is 52 and the second highest 45. Implicitly, result B is not considered better than A, because its high score is lower than that of A. But neither is result A considered superior, because its second high score is lower than that of B.

One way to rank all results is to consider only the rth highest score and calculate, for instance, the probability that the second best score is S or greater. In the above example, result B would be considered superior to A because its second best score is the greater (see Fig. 2). In the no-gap case, the probability that the rth highest scoring locally optimal subalignment has score at least S is derived from the Poisson distribution,[8] and it is well approximated by

$$1 - e^{-a}\sum_{i=0}^{r-1}\frac{a^i}{i!} \tag{6}$$

where $a = Kmn\ e^{-\lambda S}$. For subalignments allowing gaps, simulation has shown this formula to hold extremely well, once K and λ are estimated.[17,18]

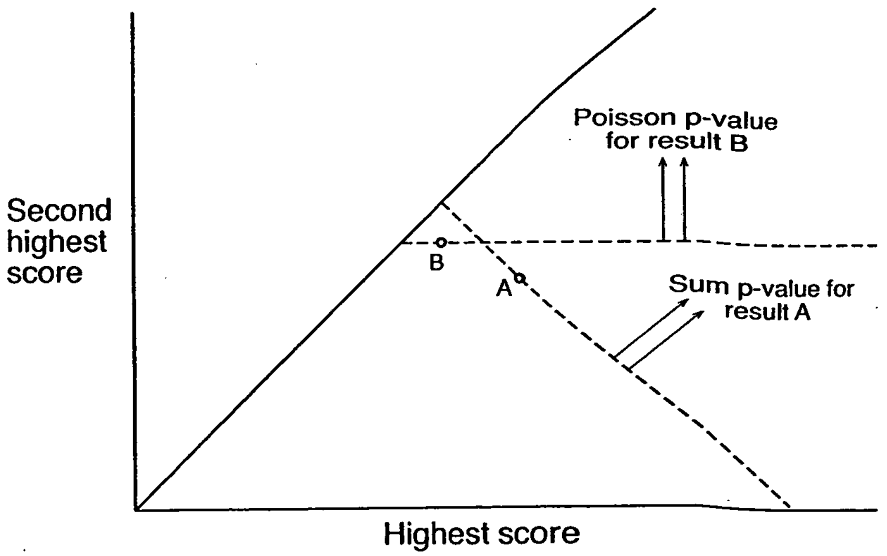


FIG. 2. The calculation of Poisson and sum p values. The region on or below the main diagonal represents feasible joint values of the first and second highest locally optimal subalignment scores. A probability density may be imagined to cover this space.[12] The probability that the second highest score is greater than or equal to 45 is that lying on or above the horizontal line through point B. The probability that the sum of the two highest scores is greater than or equal to 105 is that lying on or above the antidiagonal line through point A. Which result is considered the more significant depends on which criterion is employed.

An alternative and perhaps more natural way to assess a collection of the r highest distinct subalignment scores is by means of their sum. In the above example, result A would be considered better than B because 105, the sum of its two scores, is greater than 97 (see Fig. 2). When gaps are disallowed, the asymptotic distribution for the sum of the r highest scores from locally optimal subalignments has been derived.[12] If S_r is the rth highest score, let $T_r = \sum_{i=1}^{r} (\lambda S_i - \ln Kmn)$. Then for large mn the probability density function for T_r approaches

$$f(t) = \frac{e^{-t}}{r!(r-2)!} \int_0^\infty y^{r-2} \exp(-e^{(y-t)/r})\, dy \qquad (7)$$

To obtain the tail probability that $T_r \geq x$ one must integrate $f(t)$ from Eq. (7) for t from x to infinity.[12] This double integral is easily calculated numerically, and a program for the purpose in the C programming language is available from the authors.

Here, we wish to investigate whether the sum statistics described by Eq. (7) are applicable as well to subalignments with gaps. To test this hypothesis, we generated 10,000 pairs of random protein sequences of length 600. Using BLOSUM62 substitution scores and (12, 1) gap costs, we collected for each sequence pair the 10 best locally optimal subalignment scores. We estimated λ and K from the optimal scores by the method of moments. We then tabulated, for r from 1 to 10, the frequency with which the sum of the r highest scores achieved various values, and calculated from Eq. (7) the expected number of times these values should have been attained. The results for $r = 5$ are shown in Fig. 3 and Table III, and Table III also shows a χ^2 goodness-of-fit calculation. Because λ and K were estimated from the data, the number of degrees of freedom in this example is 71 and the χ^2 statistic is 89.7, not surprisingly high under the null hypothesis that the theory is exactly valid. Table IV gives χ^2 values for all r tested, along with the scores characterizing several calculated and observed significance levels. It is evident that the sum statistics perform quite well: although the χ^2 goodness-of-fit test evinces imperfect conformity with the theory, the 95, 99, and 99.9% significance levels are predicted within an error of ± 1 in almost every case. Similar or better results were obtained for other substitution matrices and gap costs. We note that the gap costs were small enough, in this example, for over one-half the optimal subalignments to contain gaps, so that we are not simply verifying the theory for the no-gap case, where it is known to hold.

We conclude that the statistical theory for ungapped alignments carries over essentially unchanged to gapped alignments. All that is required for its application are good estimates of λ and K and, if edge corrections are desired, H as well. The statistical significance of optimal subalignment

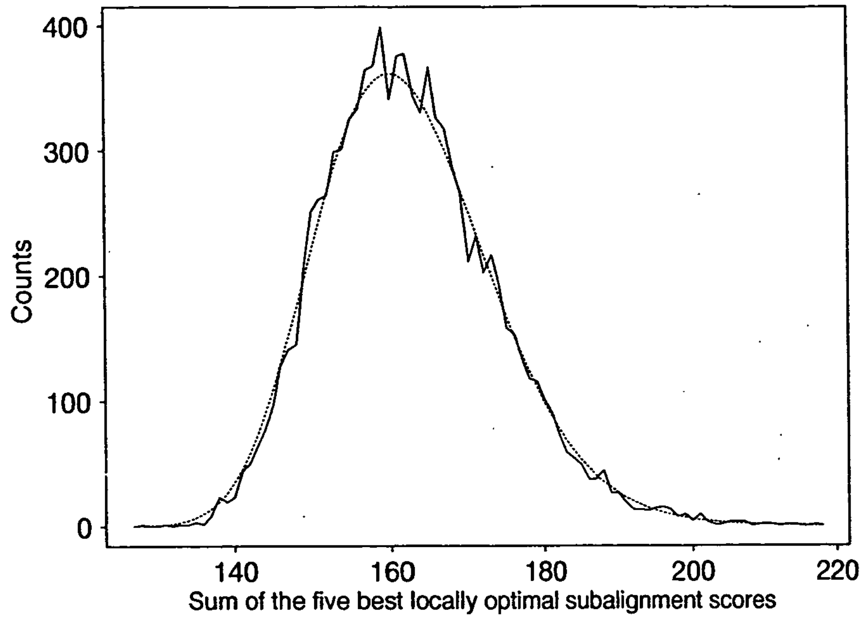


FIG. 3. Expected and observed counts, from 10,000 random trials, for the sum of the five highest locally optimal subalignment scores. The dotted line represents the theoretical distribution, based on values of λ and K estimated from the data.[12] The solid line connects data points from the observed distribution (see Table III). The χ^2 goodness-of-fit statistic (with 71 degrees of freedom) is 89.7. Even were the theory precisely valid, a worse fit would be obtained in about 7% of all stochastic experiments. Curves with this or a better fit to theory were obtained for five of the ten values of r tested (see Table IV).

scores, and of the sum of locally optimal subalignment scores, is then readily calculated.

On a practical note, one rarely knows *a priori* whether the criterion one uses for assessing a given comparison of two sequences should be the highest subalignment score, or the sum of the two highest, or the sum of the three highest, etc. One way to decide is to choose the collection of subalignments that yields by Eq. (7) the smallest p value. A difficulty is that one has then performed multiple tests, and the p value one had calculated is accordingly no longer valid. A simple and conservative remedy has been suggested by P. Green and implemented in the BLAST programs.[3] The basic idea is to use a geometric series in r to discount the significance of results involving r scores.[38] The same strategy may be applied when the Poisson statistics of Eq. (6) are used in place of sum statistics.

It is interesting to compare the joint evaluation of multiple independent subalignments by means of sum statistics, and the consolidation of these subalignments using gaps and their attendant costs. Sum statistics allow gaps of arbitrary length, in either or both sequences, between the subalignments

[38] S. F. Altschul, M. S. Boguski, W. Gish, and J. C. Wootton, *Nat. Genet.* 6, 119 (1994).

TABLE III
STOCHASTIC EXPERIMENT FOR SUM OF FIVE BEST LOCALLY OPTIMAL SUBALIGNMENT SCORES[a]

Sum of scores	Expected number	Observed number	χ^2	Sum of scores	Expected number	Observed number	χ^2
<133	4.0	2	1.00	170	250.3	211	6.16
133, 134	6.9	2	3.48	171	233.0	231	0.02
135	6.4	3	1.78	172	215.7	202	0.87
136	9.4	1	7.50	173	198.7	216	1.51
137	13.5	8	2.23	174	182.1	191	0.43
138	18.9	23	0.90	175	166.1	158	0.40
139	25.8	19	1.80	176	150.9	153	0.03
140	34.4	23	3.79	177	136.4	133	0.09
141	45.1	44	0.03	178	122.8	117	0.28
142	57.7	50	1.03	179	110.2	115	0.21
143	72.5	64	0.99	180	98.4	101	0.07
144	89.3	77	1.70	181	87.6	90	0.07
145	108.1	96	1.36	182	77.6	73	0.28
146	128.7	127	0.02	183	68.6	59	1.35
147	150.7	141	0.62	184	60.4	54	0.68
148	173.7	145	4.75	185	53.0	49	0.31
149	197.4	206	0.38	186	46.4	37	1.91
150	221.1	251	4.03	187	40.5	37	0.30
151	244.5	261	1.12	188	35.2	44	2.19
152	266.9	264	0.03	189	30.6	26	0.68
153	287.8	299	0.44	190	26.4	26	0.01
154	306.7	301	0.11	191	22.8	19	0.63
155	323.3	325	0.01	192	19.6	13	2.23
156	337.2	333	0.05	193	16.8	13	0.88
157	348.2	365	0.81	194	14.4	12	0.41
158	356.0	368	0.40	195	12.3	14	0.23
159	360.7	399	4.07	196	10.5	15	1.93
160	362.2	341	1.24	197	8.9	13	1.85
161	360.6	376	0.65	198	7.6	7	0.04
162	356.2	378	1.34	199	6.4	9	1.03
163	349.1	344	0.07	200	5.4	4	0.38
164	339.6	330	0.27	201	4.6	9	4.25
165	327.8	367	4.69	202	3.9	3	0.19
166	314.8	326	0.40	203, 204	6.0	2	2.67
167	299.9	317	0.98	205, 206	4.2	6	0.77
168	284.0	286	0.01	207–210	4.9	6	0.25
169	267.4	266	0.01	>210	4.4	4	0.04

[a] Ten thousand random protein sequence pairs of length 600 were generated, using amino acid frequencies from Robinson and Robinson.[27] Subalignment scores were calculated using (12, 1) gap costs and the BLOSUM62 matrix.[28]

TABLE IV

SIGNIFICANCE LEVELS AND χ^2 STATISTICS FOR SUM OF THE r BEST LOCALLY OPTIMAL SUBALIGNMENT SCORES

	Estimated/observed significance levels					
r	95%	99%	99.9%	χ^2 statistic	Degrees of freedom	P value (%)
1	47/47	53/54	62/63	42.3	32	10
2	85/84	92/92	102/103	59.1	45	8
3	119/119	128/128	139/140	61.9	54	20
4	152/152	162/162	174/175	62.3	63	50
5	184/183	194/195	207/207	89.7	71	7
6	214/214	226/226	239/239	117.2	77	0.2
7	244/244	256/256	271/270	111.7	83	2
8	274/273	286/286	301/301	111.5	88	5
9	302/301	316/314	332/332	121.4	94	3
10	331/330	344/343	361/362	135.7	99	0.8

involved, and in fact do not require that these subalignments be combinable into a consistent larger subalignment. (The statistics may be sharpened by requiring consistent ordering of the subalignments.)[12] Each such gap is paid for by an increase in r, and thus an extra subtraction of ln Kmn in the calculation of T_r. The net effect is, to first order, an effective cost of ln Kmn nats for combining subalignments. This is substantially greater than the costs normally employed for short gaps,[39] as will be seen below, but it does permit great flexibility in the combination of subalignments.

Statistical Parameters for Frequently Used Substitution Matrices

For reference purposes, we have compiled in Tables V–VII statistical parameters for several frequently used amino acid substitution matrices. Several warnings should be given concerning the accuracy of these tables. First, Tables V–VII were constructed using a specific amino acid frequency model,[27] which may be poor for any particular pair of proteins. Mott has attempted to identify systematic dependencies of statistical parameters on amino acid composition,[16] and this may prove a fruitful direction for further research. Second, except those for infinite gap costs, which were derived from theory, all parameter values are stochastically determined and therefore involve random error. Finally, gap costs which involve relative entropies less than about 0.15 nats have optimal subalignments with average

[39] W. R. Pearson, *Protein Sci.* **4,** 1145 (1995).

TABLE V
STATISTICAL PARAMETERS FOR (a, b) AFFINE GAP COSTS IN CONJUNCTION WITH BLOSUM62 SUBSTITUTION SCORES[a]

a	b	λ	K	H (nats)	a	b	λ	K	H (nats)
∞	0–∞	0.318	0.13	0.40	8	7, 8	0.270	0.06	0.25
					8	4–6	0.262	0.05	0.23
12	3–12	0.305	0.10	0.38	8	3	0.243	0.035	0.18
12	2	0.300	0.09	0.34	8	2	0.215	0.021	0.12
12	1	0.275	0.05	0.25	8	1		Borderline	
11	3–11	0.301	0.09	0.36	7	6, 7	0.247	0.05	0.18
11	2	0.286	0.07	0.29	7	4, 5	0.230	0.030	0.15
11	1	0.255	0.035	0.19	7	3	0.208	0.021	0.11
					7	2	0.164	0.009	0.06
10	4–10	0.293	0.08	0.33	7	1		Linear	
10	3	0.281	0.06	0.29					
10	2	0.266	0.04	0.24	6	5, 6	0.200	0.021	0.10
10	1	0.216	0.014	0.12	6	4	0.179	0.014	0.08
					6	3	0.153	0.010	0.05
9	5–9	0.286	0.08	0.29	6	1, 2		Borderline or linear	
9	3, 4	0.273	0.06	0.25					
9	2	0.244	0.030	0.18	5	5	0.131	0.009	0.04
9	1	0.176	0.008	0.06	1–5	1–4		Borderline or linear	

[a] Parameters for infinite gap costs were calculated using the analytic theory.[8] Linear designates scoring systems in the linear as opposed to the logarithmic domain, where the extreme value theory does not apply.[13] Borderline designates scoring systems near the logarithmic–linear phase transition.[13] From Ref. 28.

length greater than 50. This is a large fraction of the length of the random sequences generated, and thus edge effects begin to be substantial. The most important effect, suggested by the trends seen in Table I, may be an overestimate of the asymptotic value of λ.

It should furthermore be understood that both the PAM[29,30]and BLOSUM[28] matrices consist of log-odds scores constructed explicitly from target frequencies for aligned amino acid pairs. The theory that supports the use of such scores[20] breaks down once gaps are allowed. Thus, although nothing prevents these matrices from being used in conjunction with gap scores, substitution scores constructed in some other manner may be preferable. The practice of subtracting some value from all matrix elements,[16] for example, has sometimes been advocated. However, except in broad outline, theory remains silent on the best way to choose not only gap scores, but also the substitution scores that are used with them.

There are several lessons to be drawn from the numbers in Tables V–VII. For a given gap-opening cost a, the statistical parameters do not

TABLE VI
STATISTICAL PARAMETERS FOR (a, b) AFFINE GAP COSTS IN CONJUNCTION WITH BLOSUM50 SUBSTITUTION SCORES[a]

a	b	λ	K	H (nats)
∞	0–∞	0.232	0.11	0.34
16	4–16	0.222	0.08	0.31
16	3	0.213	0.06	0.27
16	2	0.207	0.05	0.24
16	1	0.180	0.024	0.15
15	8–15	0.222	0.09	0.31
15	6, 7	0.219	0.08	0.29
15	4, 5	0.216	0.07	0.28
15	3	0.210	0.06	0.25
15	2	0.202	0.05	0.22
15	1	0.166	0.018	0.11
14	8–14	0.218	0.08	0.29
14	5–7	0.214	0.07	0.27
14	4	0.205	0.05	0.24
14	3	0.201	0.05	0.22
14	2	0.188	0.034	0.17
14	1	0.140	0.009	0.07
13	8–13	0.211	0.06	0.27
13	5–7	0.205	0.05	0.24
13	4	0.202	0.05	0.22
13	3	0.188	0.034	0.18
13	2	0.174	0.025	0.13
13	1	0.114	0.006	0.04
12	7–12	0.205	0.06	0.24
12	5, 6	0.197	0.05	0.21
12	4	0.192	0.04	0.18
12	3	0.178	0.028	0.15
12	2	0.158	0.019	0.10
12	1		Borderline	
11	8–11	0.197	0.05	0.21
11	6, 7	0.190	0.04	0.19
11	5	0.184	0.04	0.17
11	4	0.177	0.031	0.15
11	3	0.167	0.028	0.11
11	2	0.130	0.009	0.06
11	1		Linear	
10	8–10	0.183	0.04	0.17
10	6, 7	0.178	0.035	0.16
10	5	0.168	0.026	0.13
10	4	0.156	0.020	0.10
10	3	0.139	0.013	0.07
10	2	0.099	0.007	0.03
10	1		Linear	
9	7–9	0.164	0.029	0.13
9	5, 6	0.152	0.021	0.10
9	4	0.134	0.014	0.07
9	3	0.107	0.008	0.04
9	1, 2		Linear	
8	8	0.139	0.017	0.08
8	7	0.134	0.015	0.07
8	6	0.127	0.013	0.06
8	5	0.117	0.011	0.05
8	4	0.101	0.009	0.03
8	1–3		Borderline or linear	
7	7	0.100	0.010	0.04
7	6	0.094	0.010	0.03
7	1–5		Borderline or linear	
1–6	1–6		Linear	

[a] From Ref. 28.

change appreciably for gap-extension costs b greater than some value b'. This may at first seem surprising but it indicates that, with these costs, very few locally optimal subalignments that arise by chance contain gaps of length greater than one. One conclusion is that for a given a it is unrewarding to use any gap extension penalty greater than b'. The noise from high-

TABLE VII
STATISTICAL PARAMETERS FOR (a, b) AFFINE GAP COSTS IN CONJUNCTION WITH PAM250 SUBSTITUTION SCORES[a]

a	b	λ	K	H (nats)	a	b	λ	K	H (nats)
∞	0–∞	0.229	0.09	0.23	11	7–11	0.186	0.04	0.13
					11	5, 6	0.180	0.034	0.11
16	4–16	0.217	0.07	0.21	11	4	0.165	0.021	0.09
16	3	0.208	0.05	0.18	11	3	0.153	0.017	0.07
16	2	0.200	0.04	0.16	11	2	0.122	0.009	0.04
16	1	0.172	0.018	0.09	11	1		Linear	
15	5–15	0.215	0.06	0.20	10	8–10	0.175	0.031	0.11
15	4	0.208	0.05	0.18	10	7	0.171	0.029	0.10
15	3	0.203	0.04	0.16	10	6	0.165	0.024	0.09
15	2	0.193	0.035	0.14	10	5	0.158	0.020	0.08
15	1	0.154	0.012	0.07	10	4	0.148	0.017	0.07
					10	3	0.129	0.012	0.05
14	6–14	0.212	0.06	0.19	10	1, 2		Borderline or linear	
14	4, 5	0.204	0.05	0.17					
14	3	0.194	0.035	0.14	9	7–9	0.151	0.020	0.07
14	2	0.180	0.025	0.11	9	6	0.146	0.019	0.06
14	1	0.131	0.008	0.04	9	5	0.137	0.015	0.05
					9	4	0.121	0.011	0.04
13	6–13	0.206	0.06	0.17	9	3	0.102	0.010	0.03
13	4, 5	0.196	0.04	0.14	9	1, 2		Linear	
13	3	0.184	0.029	0.12					
13	2	0.163	0.016	0.08	8	7, 8	0.123	0.014	0.05
13	1	0.110	0.008	0.03	8	6	0.115	0.012	0.04
					8	5	0.107	0.011	0.03
12	7–12	0.199	0.05	0.15	8	1–4		Borderline or linear	
12	5, 6	0.191	0.04	0.13					
12	4	0.181	0.029	0.12	7	7	0.090	0.014	0.02
12	3	0.170	0.022	0.10	7	1–6		Borderline or linear	
12	2	0.145	0.012	0.06					
12	1		Borderline		1–6	1–6		Linear	

[a] From Refs. 29 and 30.

scoring random subalignments will not decrease appreciably, but the occasional true subalignment with a long gap will be penalized.

For a given set of gap costs, it is also useful to consider the ratio of λ to λ_∞ for infinite gap costs. This ratio indicates the proportion of information in ungapped alignments that must be sacrificed in the hope of extending the alignments using gaps. For example, (12, 1) gap costs used in conjunction with BLOSUM62 scores sacrifice about 14% of the information in ungapped alignments, reducing, for example, an alignment with a score of 30 nats to one of about 26 nats. However, gap costs frequently allow ungapped alignments to be extended, recouping in the process more than the amount of information lost. The best gap costs will be those that on average maximize the difference between information gained and information lost.

Fairly low gap costs are sometimes employed so that subalignments involving long insertions or deletions will not be too severely penalized. This, however, decreases the value of any score obtained by lowering the value of λ. (An alternative view is that it raises the noise level by permitting the appearance of higher scoring random alignments.) One possible compromise is to charge fairly high gap penalties, but to evaluate the appearance of multiple high-scoring locally optimal subalignments using the sum statistics described above. This allows small gaps to be incorporated without unduly deflating the currency of alignment score, but it also permits gaps of arbitrary size to contribute to the significance of a result. Although such ideas may inform the search, the choice of appropriate gap costs still relies on empiricism.[39,40]

Empirical Statistics from Database Searches

In many situations, searching protein or DNA sequence databases for local alignments using the Smith–Waterman algorithm[1] or its variations[31–34] requires too much time to be practical. Accordingly, parallel architecture computers,[41–45] specialized VLSI chips,[46–48] and heuristic algorithms[2,3,49] have been emplyed for the purpose. In the examples below, we use a heuristic algorithm BLASTGP, which is a new version of the BLAST programs that permits gaps in protein sequence comparisons. This experi-

[40] G. Vogt, T. Etzold, and P. Argos, *J. Mol. Biol.* **249,** 816 (1995).

[41] A. F. W. Coulson, J. F. Collins, and A. Lyall, *Comput. J.* **30,** 420 (1987).

[42] R. Jones, *CABIOS* **8,** 377 (1992).

[43] G. Vogt and P. Argos, *CABIOS* **8,** 49 (1992).

[44] D. L. Brutlag, J.-P. Dautricourt, R. Diaz, J. Fier, B. Moxon, and R. Stamm, *Comput. Chem.* **17,** 203 (1993).

[45] S. S. Sturrock and J. F. Collins, "MPsrch Version 1.3." Biocomputing Research Unit, University of Edinburgh, 1993.

[46] E. T. Chow, T. Hunkapiller, J. C. Peterson, B. A. Zimmerman, and M. S. Waterman, *in* "Proceedings of the 1991 International Conference on Supercomputing," p. 216. ACM Press, New York, 1991.

[47] R. P. Hughey, Ph.D. Thesis, Brown University, Providence, Rhode Island (1991).

[48] C. T. White, R. K. Singh, P. B. Reintjes, J. Lampe, B. W. Erickson, W. D. Dettloff, V. L. Chi, and S. F. Altschul, *in* "Proceedings of the 1991 IEEE International Conference Computer Design: VLSI in Computers and Processors," p. 504. IEEE Computer Society Press, Los Alamitos, California, 1991.

[49] A. Califano and I. Rigoutsos, *in* "Proceedings of the First International Conference on Intelligent Systems for Molecular Biology" (L. Hunter, D. Searls, and J. Shavlik, eds.), p. 56. AAAI Press, Menlo Park, California, 1993.

mental program is available from the authors on request.[50] Like the program LFASTA, its basic strategy is to use a banded Smith–Waterman algorithm[51] centered on high-scoring segment pairs (HSPs) identified by the BLAST strategy.[3] When all HSPs found are processed in this way, the search time is about 50% greater than that of BLASTP; when only HSPs significant in the context of the pairwise comparison in which they occur are processed, the search time is increased by about 10%. Using BLASTGP, we can study the use of sum statistics on gapped subalignments involving real sequences.

To test whether the statistics developed above are approximately valid for protein database searches, we compared shuffled versions of 1000 randomly selected proteins to the SWISS-PROT protein sequence database, Release 31.0.[52] Before searching, each shuffled sequence was filtered using the SEG program[38,53] to remove regions of highly biased amino acid composition; such regions may arise even in shuffled sequences when the source sequence begins with a biased composition. The database searches were performed using the BLOSUM62 substitution matrix[28] and (12, 2) affine gap costs. The corresponding values of $\lambda = 0.300$ and $K = 0.09$ from Table V were used to calculate sum statistic p values for any set of locally optimal subalignments found for a given database sequence. A p value cutoff was imposed so that 10 sequences were expected to be found in each database search.[38] For the 1000 trials, the mean number of sequences reported was 13.9, and the median number 8, in good agreement with prediction. For a comparable 1000 BLASTP searches, in which the statistical parameters were calculated analytically, the mean and median number of sequences found were 10.6 and 7.

Biological Example

To illustrate several potential effects of gap costs and sum statistics, we consider a database search of SWISS-PROT Release 31[52] (43,470 sequences; 15.3 million residues) with a hypothetical yeast protein of length 234 (SWISS-PROT accession number P40582)[54] as query. Using the BLASTGP program in conjunction with BLOSUM62 substitution scores and (12, 2) affine gap costs, two statistically significant similarities are found. The highest score for a single subalignment (Fig. 4a) is 74 (32.0 bits[20]) and involves a *Silene cucubalus* glutathione *S*-transferase (SWISS-PROT accession num-

[50] Write Warren Gish (gish@watson.wustl.edu).

[51] K.-M. Chao, W. R. Pearson, and W. Miller, *CABIOS* **8**, 481 (1992).

[52] A. Bairoch and B. Boeckmann, *Nucleic Acids Res.* **22**, 3578 (1994).

[53] J. C. Wootton and S. Federhen, *Comput. Chem.* **17**, 149 (1993).

[54] B. G. Barrel *et al.*, unpublished.

```
(a)

P40582      6 IKVHWLDHSRAF-RLLWLLDHLNLEYEIVPYKRDANFRAPPELKKIHPLGRSPLLE  60
              IKVH    S A  R+L  L   +LE+E VP    A     P    ++P G+ P LE
Q04522      2 IKVHGNPRSTATQRVLVALYEKHLEFEFVPIDMGAGGHKQPSYLALNPFGQVPALE  57
                           *******************************************

           61 VQDRETGKKKILAESGFIFQYVLQHFDHSH  90
                    G+ K+  ES  I +Y+    DH +
           58 -----DGEIKLF-ESRAITKYLAYTHDHQN  81


(b)

P40582     35 YKRDANFRAPPELKKIHPLGRSPLLEVQDR  64
              Y+  A+ R  P LK   P+G+ P+LEV  +
P41043     74 YEDVAHPRRVPALKPTMPMGQMPVLEVDGK 103
              ******************************


P40582    118 LMIEFILSKVKDSGMPFPI-SYLAR-KVADKISQAYSSGEVKNQFDFVEGEISKNN 171
              L I+ ++  + D  +   + SY    ++ +K     ++  +    + +E  +  N+
P41043    130 LQIDIVVDTINDFRLKIAVVSYEPEDEIKEKKLVTLNAEVIPFYLEKLEQTVKDND 185
                                                               *********

          172 GYLVDGKLSGADILMS-FPLQMAFERKFAAPEDYPAISKWLKTITSEESYAASKEK 226
              G+L  GKL+ AD+  +     M +  K    E YPA+   +  + + E   A  EK
          186 GHLALGKLTWADVYFAGITDYMNYMVKRDLLEPYPAVRGVVDAVNALEPIKAWIEK 241
              **************


(c)

P40582     18 RLLWLLDHLNLEYEIVPYKRDANFRAPPELKKIHPLGRSPLLEVQDRETGKKKILA  73
              R+  +L+   L++EIVP          P+   ++P G+ P L   D      ++L
P04907     16 RVATVLNEKGLDFEIVPVDLTTGAHKQPDFLALNPFGQIPALVDGD------EVLF  65
              ******************************************

           74 ESGFIFQYVLQHF  86
              ES  I +Y+   +
           66 ESRAINRYIASKY  78
```

FIG. 4. Locally optimal subalignments from the comparison of a hypothetical yeast protein (SWISS-PROT accession number P40582),[54] with various sequences in SWISS-PROT Release 31.[52] Alignments were found using the program BLASTGP, with BLOSUM62[28] amino acid substitution costs and (12, 2) affine gap costs. Central lines of alignments echo identities, and plus symbols (+) indicate positions with positive substitution score. Sections of alignments underlined with asterisks are ungapped alignments found by BLASTP[3] using the same substitution costs. (a) Locally optimal subalignment with score 74 and E value 0.045 involving an *S. cucubalus* glutathione *S*-transferase sequence (accession number Q04522).[55] When gaps are disallowed, the underlined region has score 62 and E value 0.90. (b) Two locally optimal subalignments, with respective scores 54 and 57, involving a *Drosophila* glutathione *S*-transferase sequence (accession number P41043).[56] The sum statistic E value[12,38] for the two subalignments is 0.014. When gaps are disallowed, the score of the underlined region in the second subalignment is 47, and the combined E value is 0.096. (c) Locally optimal subalignment with score 59 and E value 4.8 involving a maize glutathione *S*-transferase III sequence (accession number P04907).[57] When gaps are disallowed, the underlined region has score 58 and E value 3.5.

ber Q04522).[55] In the context of the search,[38] the expected number of occurrences (E value)[38] of a subalignment at least as good as this one is 0.045. If no gaps are allowed, the subalignment shrinks to the segment pair with score 62 (28.6 bits) indicated in Fig. 4a, whose E value of 0.9 is twenty times larger.

An even more significant result involves the two locally optimal subalignments with a *Drosophila* glutathione *S*-transferase sequence (SWISS-PROT accession number P41043)[56] shown in Fig. 4b. Neither subalignment, the first with score 54 (23.4 bits) and the second with score 57 (24.7 bits), would by itself be significant in the context of a database search.[38] However, using the sum statistics described above,[12] the two alignments in conjunction are equivalent to a single alignment of 33.7 bits, which here has an E value of 0.014. When no gaps are allowed, the second subalignment is trimmed to the indicated segment pair, with score 47 (21.7 bits), shown in Fig. 4b; the E value for the combined segment pairs increases by a factor of seven. The subalignments shown would be joined were lower gap costs used, but this would decrease λ and thereby dilute the value of the nominal score attained.

Not every alignment, of course, is rendered more significant by allowing gaps. A locally optimally subalignment of the query with a maize glutathione *S*-transferase III sequence (SWISS-PROT accession number P04907)[57] is shown in Fig. 4c. This alignment has score 59 (25.5 bits), whereas the one to which it shrinks when gaps are disallowed has score 58 (26.7 bits). Although the nominal score increases with the introduction of gaps, the 6% decrease in λ (Table V) renders the score less surprising.

The substitution matrix and gap costs employed in this example are not necessarily the best for recognizing distant similarities[39,40] and are used only for illustration. The program BLASTGP is similar in many ways to more recent experimental versions of FASTA, and a thorough comparison[39] of their relative speeds and sensitivities awaits further study.

Acknowledgments

We thank Dr. Martin Vingron for helpful conversions, and Dr. Gregory Schuler for assistance in the production of Figure 2.

[55] T. M. Kutchan and A. Hochberger, *Plant Physiol.* **99,** 789 (1992).
[56] C. Beall, C. Fyrberg, S. Song, and E. Fyrberg, *Biochem. Genet.* **30,** 515 (1992).
[57] R. E. Moore, M. S. Davies, K. M. O'Connell, E. I. Harding, R. C. Wiegand, and D. C. Tiemeier, *Nucleic Acids Res.* **14,** 7227 (1986).

[28] Parametric and Inverse-Parametric Sequence Alignment with XPARAL

By D. GUSFIELD and P. STELLING

Introduction

When aligning DNA or amino acid sequences using numerical-based optimization, there is often considerable disagreement about how to weight matches, mismatches, insertions and deletions (indels), and gaps. Most alignment methods require the user to specify fixed values for those parameters, and it is widely observed that the quality of the resulting alignment can be greatly affected by the choice of parameter settings.

Parametric alignment attempts to avoid the problem of choosing fixed parameter settings by computing the optimal alignment as a function of variable parameters for weights and penalties. The goal is to partition the parameter space into regions (which are necessarily convex) such that in each region one alignment is optimal throughout and such that each region is maximal for this property. Thus parametric alignment allows one to see explicitly, and completely, the effect of parameter choices on the optimal alignment. Parametric sequence alignment was first used in a paper by Fitch and Smith[1] and was studied more extensively in papers by Gusfield *et al.*,[2,3] Waterman *et al.*,[4] and Vingron and Waterman.[5] The last four papers concern the number, shape, pattern, and interpretation of the regions. The basic algorithmic ideas for computing two-dimensional parametric decompositions were first developed in contexts other than sequence alignment.[6]

In this chapter we first describe a publicly available, user-friendly interactive software package, XPARAL, that solves the parametric alignment problem; we emphasize newer features in XPARAL. We next illustrate the use of XPARAL by reexamining a study done by Barton and Sternberg[7] on gap weights used in aligning protein secondary structure. Finally, we discuss the empirical and theoretical efficiency of XPARAL. We use stan-

[1] W. Fitch and T. Smith, *Proc. Natl. Acad. Sci. U.S.A.* **80,** 1382 (1983).
[2] D. Gusfield, K. Balasubramonian, and D. Naor, "Proceedings of the Third Annual Symposium on Discrete Algorithms," p. 432. Orlando, FL. 1992.
[3] D. Gusfield, K. Balasubramonian, and D. Naor, *Algorithmica* **12,** 312 (1994).
[4] M. Waterman, M. Eggert, and E. Lander, *Proc. Natl. Acad. Sci. U.S.A.* **89,** 6090 (1992).
[5] M. Vingron and M. Waterman, *J. Mol. Biol.* **235,** 1 (1994).
[6] D. Gusfield, *J. ACM* **30,** 551 (1983).
[7] G. Barton and M. Sternberg, *Protein Eng.* **1,** 89 (1987).